ulm university universität ulm

**Universität Ulm** | 89069 Ulm | Germany

**Fakultät für Ingenieurwissenschaften und Informatik**
Institut für Datenbanken und Informationssysteme

# Visualisierung und Navigation komplexer Produktdaten

Bachelorarbeit an der Universität Ulm

**Vorgelegt von:**
Stefan Hausladen
stefan.hausladen@uni-ulm.de

**Gutachter:**
Prof. Dr. Manfred Reichert

**Betreuer:**
Dipl.-Inf. Julian Tiedeken

2015

„Visualisierung und Navigation komplexer Produktdaten“
Fassung vom 16. Februar 2015


Satz: PDF-LaTeX 2ε

# Inhaltsverzeichnis

**Abstract** **1**

**1 Einleitung** **3**

1.1 Motivation . . . . . 4

1.2 Aufgabenstellung . . . . . 4

1.3 Aufbau der Arbeit . . . . . 5

**2 Das PROCEED-Framework** **7**

2.1 Problemstellung . . . . . 7

2.2 Bestandteile von PROCEED . . . . . 8

2.2.1 Produktdatenapplikation . . . . . 8

2.2.2 Produktdatenebenen . . . . . 9

2.2.3 Local Ontologies . . . . . 9

2.2.4 Common Integration Ontology . . . . . 11

2.3 Visualisierung und Navigation von Datenabfragen . . . . . 14

**3 Grundlagen des Usability Engineerings** **17**

3.1 Bedeutungen und Funktionen eines Bildes . . . . . 17

3.2 Medienspezifische Lerntheorien . . . . . 18

3.3 Grundlagen der Bildgestaltung aus psychologischer Sicht . . . . . 20

3.3.1 Nutzung eines Bildes . . . . . 20

3.3.2 Einschränkungen durch die Arbeitsgedächtniskapazität . . . . . 21

3.4 Einschränkungen des Begriffs „visueller Ansatz“ . . . . . 22

3.5 Usability Engineering . . . . . 22

3.6 Benutzerschnittstellen . . . . . 24

3.6.1 Ein-/Ausgabeschnittstelle . . . . . 25

3.6.2 Dialogschnittstelle . . . . . 25

3.6.3 Werkzeugschnittstelle . . . . . 25

3.7 Gestaltungsgrundsätze . . . . . 26
3.8 Allgemeines Vorgehen bei der Gestaltung . . . . . 27
3.9 Das Referenzmodell für Usability Engineering der Daimler AG . . . . . 29

**4 Anforderungsanalyse 37**
4.1 Geschäfts-/Einsatzziele . . . . . 37
4.2 Analyse Ist-Stand . . . . . 38
4.3 Benutzerprofilanalyse . . . . . 39
4.4 Kontextuelle Aufgabenanalyse . . . . . 40
4.5 Umgebungsbedingungen . . . . . 43

**5 Übertragung der Systemanforderungen auf die Gestaltung 45**
5.1 Anforderung 1: Darstellung der einzelnen abgefragten Objekte aus der CIO . 46
5.2 Anforderung 2: Ansicht der Artefaktattribute . . . . . 47
5.3 Anforderung 3: Geeignete Gesamtdarstellung aller Ergebniselemente . . . . 49
5.3.1 Geordnete Struktur . . . . . 50
5.3.2 Geometrische Darstellungsform . . . . . 52
5.3.3 Ungeordnete Darstellung . . . . . 53
5.4 Anforderung 4: Geeigneter Rahmen für die Gesamtdarstellung . . . . . 54
5.4.1 Verwendung von Fenstern . . . . . 55
5.4.2 Verwendung von Tabs . . . . . 55
5.4.3 Verwendung von Seiten . . . . . 56
5.5 Anforderung 5: Korrespondierende Elemente aus Local Ontologies berücksichtigen . . . . . 57
5.5.1 Darstellungsort . . . . . 58
5.5.2 Darzustellende Informationen . . . . . 59
5.5.3 Gesamtdarstellung . . . . . 62
5.5.4 Darstellung der Elemente standardmäßig oder auf Wunsch . . . . . 63
5.5.5 Zusätzliche Möglichkeiten bei elementweiser Darstellung . . . . . 64
5.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen darstellen . . . . . 66
5.5.7 Berücksichtigung von mehrwertigen Korrespondenzen . . . . . 69
5.6 Anforderung 6: Ebenenwechsel zu Varianten und Versionen . . . . . 70
5.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten 71
5.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen . . . . . 71

5.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen 72
Fenster . . . 72
Tabs . . . 73
Seitenansicht . . . 74
5.6.4 Ebenenwechsel - global oder elementweise . . . 76
Globaler Ebenenwechsel . . . 76
Elementweiser Ebenenwechsel . . . 77
5.6.5 Ebenen der korrespondierenden Elemente berücksichtigen . . . 79

**6 Kompletter elementweiser Ansatz 85**
6.1 Darstellung der einzelnen abgefragten Objekte aus der CIO . . . 85
6.2 Ansicht der Artefaktattribute . . . 86
6.3 Geeignete Gesamtdarstellung aller Ergebniselemente . . . 88
6.4 Geeigneter Rahmen für eine Gesamtdarstellung . . . 89
6.5 Darstellung der korrespondierenden Elemente aus Local Ontologies . . . 90
6.5.1 Darstellungsort . . . 90
6.5.2 Dargestellte Informationen . . . 90
6.5.3 Gesamtdarstellung . . . 91
6.5.4 Zeitpunkt der Darstellung . . . 92
6.5.5 Darstellungsart . . . 92
6.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen . 94
6.5.7 Berücksichtigung von mehrwertigen Korrespondenzen . . . 94
6.6 Ebenenwechsel . . . 96
6.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten 96
6.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen . . . 96
6.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen 97
6.6.4 Art des Ebenenwechsels . . . 97
6.6.5 Ebenen der korrespondierenden Elemente . . . 98

**7 Kompletter globaler Ansatz 101**
7.1 Darstellung der einzelnen abgefragten Objekte aus der CIO . . . 101
7.2 Ansicht der Artefaktattribute . . . 101
7.3 Geeignete Gesamtdarstellung aller Ergebniselemente . . . 102
7.4 Geeigneter Rahmen für eine Gesamtdarstellung . . . 102

7.5 Darstellung der korrespondierenden Elemente aus Local Ontologies . . . . . 104
7.5.1 Darstellungsort . . . . . 104
7.5.2 Dargestellte Informationen . . . . . 105
7.5.3 Gesamtdarstellung . . . . . 105
7.5.4 Zeitpunkt der Darstellung . . . . . 106
7.5.5 Darstellungsart . . . . . 106
7.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen . 107
7.5.7 Berücksichtigung von mehrwertigen Korrespondenzen . . . . . 108
7.6 Ebenenwechsel . . . . . 108
7.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten 110
7.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen . . . . . 110
7.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen 112
7.6.4 Art des Ebenenwechsels . . . . . 112
7.6.5 Ebenen der korrespondierenden Elemente . . . . . 113

**8 Realisierung und Evaluation 115**
8.1 Elektronische UI-Prototypen . . . . . 116
8.2 Evaluationen und Tests . . . . . 121

**Zusammenfassung 123**

**Literaturverzeichnis 129**

# Abstract

Heutige Fahrzeuge bestehen aus bis zu 120 elektronischen und elektrischen Komponenten. Durch die steigende Anzahl solcher Komponenten steigt automatisch auch die Anzahl an zu verwaltenden Produktdaten. Da diese in ihrer Entwicklung aktuell noch sehr individuell behandelt und gespeichert werden, ist eine einheitliche Integration dieser Daten nur mit sehr großem Aufwand möglich. Mit Hilfe des PROCEED-Frameworks[1] soll dieses Problem gelöst werden. Durch die Schaffung von einer einheitlichen semantischen Datenstruktur soll sowohl eine anwendungsunabhängige Datenintegration als auch eine einheitliche Datenabfrage ermöglicht werden. Diese Arbeit beschäftigt sich mit der Frage, wie Ergebnisse solcher Datenabfragen visuell dargestellt werden können und wie man durch solche Ergebnisse navigieren kann. Dabei wird vorab auf die psychologischen Hintergründe von Visualisierungen und das Konzept des Usability Engineerings eingegangen. Dieses Konzept dient als Grundlage für die Visualisierung und Navigation der komplexen Produktdaten. Nachdem die für diese Arbeit relevanten Phasen des Usability Engineerings analysiert wurden, werden Anforderungen für die Visualisierung von komplexen Produktdaten abgeleitet. Auf Basis dieser Anforderungen werden in dieser Arbeit zwei komplette visuelle Ansätze entwickelt und als digitale Mockups umgesetzt.

[1] PROactive Consistency for E/E-Product-Data-Management

# 1 Einleitung

Während Fahrzeuge in den 1980er Jahren überwiegend aus mechanischen Komponenten bestanden, deren einzige elektronische Komponenten Scheibenwischer, Scheinwerfer und wenn möglich ein Radio waren, so bestehen heutige Fahrzeuge aus bis zu 120 elektronischen und elektrischen (E/E-) Komponenten [Kar13, FGH+09]. Neben Karosseriebelangen und umweltfreundlicheren Faktoren sind die E/E-Komponenten ein Hauptentwicklungspunkt. Gemeint sind damit neben Steuergeräten, Aktuatoren und Sensoren für Antrieb und Fahrwerk vor allem auch Komponenten der Komfortelektronik wie beispielsweise Klima, Infotainment, Telekommunikation sowie Fahr-, Brems- und Abstands-Assistenten. Dabei werden bereits vorhandene Komponenten stetig weiterentwickelt und immer wieder neue, innovative Komponenten in das Fahrzeug integriert. Dadurch steigt der Grad der Komplexität beziehungsweise insbesondere die Anzahl an zu verwaltenden Produktdaten innerhalb eines Autos deutlich an. Doch viel entscheidender als die hohe Anzahl an Produktdaten ist die Art und Weise, wie diese Produktdaten in ihrer Entwicklung behandelt werden. Die Entwicklung findet heutzutage nicht mehr zentral an einem Standort unter der Leitung einer Abteilung statt, sondern die Daten werden von mehreren Produzenten erstellt und von mehreren Abteilungen verarbeitet und weitergereicht. Statt in einer zentralen und vor allem einheitlichen Applikation, werden die Produktdaten in unterschiedlichen heterogenen IT Systemen dokumentiert, die untereinander keine Schnittstelle haben. Zusätzlich laufen dabei viele der Entwicklungsschritte unabhängig und teils auch parallel voneinander ab. Gerade aus diesen genannten Punkten muss dringend eine Vereinheitlichung stattfinden, damit ein Austausch beziehungsweise eine Integration von Produktdaten erfolgen kann. Schließlich soll somit eine verständlichere und klarere Informationsbereitstellung gewährleistet werden.

## 1.1 Motivation

Der Wunsch nach einer Vereinheitlichung der Datenstruktur zur Ermöglichung einer Integration von E/E-Produktdaten bedarf eines neuartigen Frameworks. Bisherige Ansätze weisen leider noch erhebliche Probleme auf, die mit Hilfe des PROCEED-Frameworks aufgegriffen und gelöst werden. PROCEED soll eine einheitliche Datenabfrage über Produktdaten aus heterogenen IT Systemen ermöglichen, welche bisher aufgrund von z.B. fehlender Dokumentation oder Mehrdeutigkeiten bei der Benennung gleicher Teile nur mit viel Aufwand möglich war. Mit diesem Framework sollen Benutzer nicht nur in der Lage sein Abfragen einfacher zu erstellen, sondern diese auch übersichtlich und grafisch an das System stellen zu können. Doch eine Unterstützung hinsichtlich der Datenabfrage für den Benutzer ist nur dann wirklich hilfreich, wenn er auch mit dem Ergebnis sinnvoll arbeiten beziehungsweise das Ergebnis effizient analysieren kann. PROCEED wird unter dem Aspekt des Usability-Engineerings entwickelt und zielt somit auf eine Entwicklung ab, die zu Gunsten der Benutzer ausfällt. Alle nötigen Arbeitsschritte des Benutzers sollen damit so effektiv und effizient wie möglich vollzogen werden können. Folglich ist es im Sinne der gesamten Entwicklung, dass auch die Ergebnisse von Abfragen dem Benutzer in einer Art und Weise dargestellt werden, die zum schnellen Verständnis beiträgt.

## 1.2 Aufgabenstellung

Das Ziel dieser Arbeit ist es, visuelle Ansätze zu konzipieren, die ein Ergebnis einer Datenabfrage darstellen sollen. Anhand einer vorher durchgeführten Usability-typischen Benutzer- und Aufgabenanalyse ergeben sich bestimmte Anforderungen an das Framework und dessen Gestaltung, die entscheidend in die Entwicklung der Ansätze eingebunden werden. Zusätzlich zur Visualisierung wird auch die Navigation durch einzelne Elemente ein Teil dieser Arbeit sein.

## 1.3 Aufbau der Arbeit

Zunächst beschäftigt sich Kapitel 2 mit einer ausführlicheren Darstellung des PROCEED-Frameworks, in dem die wichtigsten Begriffe und Strukturen geklärt werden. Dem folgt eine Eingrenzung des Themengebiets auf den eigentlichen Fokus dieser Arbeit, nämlich der Visualisierung und Navigation komplexer Produktdaten. In Kapitel 3 wird die Bedeutung von Bildern und deren Vorteile gegenüber einer textuellen Darstellung hervorgehoben. Außerdem werden grundlegende Begriffe des Usability Engineerings und Designkriterien definiert und erläutert. Im weiteren Verlauf dieses dritten Kapitels wird das Usability Referenzmodell der Daimler AG herangezogen und dessen Phasen und Prozessschritte erklärt. Aufbauend auf diesem Modell werden in Kapitel 4 die für diese Arbeit relevanten Prozessschritte der Analysephase des Referenzmodells abgearbeitet. Anhand der dort erhaltenen Anforderungen an das System werden diese in Kapitel 5 einzeln abgearbeitet und auf die Gestaltung übertragen. Die Kapitel 6 und 7 präsentieren komplette visuelle Ansätze, die die vorher analysierten Anforderungen berücksichtigen. Abschließend werden in Kapitel 8 digitale, interaktive Mockups der beiden Ansätze gezeigt, um den Bezug zur technischen Umsetzung herzustellen.

# 2 Das PROCEED-Framework

Das PROCEED-Framework ist ein Projekt des Instituts für Datenbanken und Informationssysteme der Universität Ulm in Kooperation mit der Daimler AG. Mithilfe dieses Frameworks soll die Produktdatenintegration im Automobilbereich verbessert werden. Der Sinn und Zweck dieses Projekts sowie dessen wichtigste Bestandteile und Strukturen werden in diesem Kapitel dargelegt.

## 2.1 Problemstellung

Das zugrunde liegende Problem der Automobilbranche lässt sich in einem Satz zusammenfassen: Produktdaten werden in heterogenen IT Systemen dokumentiert, die untereinander keine Schnittstellen haben. Grund dafür sind neben der in den Jahren drastisch gestiegenen Anzahl an Steuerelementen eines Autos auch die sehr dynamischen Entwicklungsprozesse im E/E-Bereich. Mechanische Produktdatenmanagement-Systeme existieren schon sehr lange. Im E/E-Bereich hingegen entstehen ständig neue Standards und Technologien. Doch für die Dokumentation von relevanten Artefakten entstehen unabhängig voneinander Applikationen, die untereinander keine Schnittstelle besitzen. Die Idee, alle Applikationen durch ein Produktdatenmanagement-System abzulösen, ist jedoch auf Grund der Rahmenbedingungen (verschiedene Personen bei der Entwicklung, unterschiedliche Anforderungen, dynamische Entwicklungsprozesse, Änderungen) nicht möglich. Durch die enge Zusammenarbeit ist es aber dringend notwendig, Daten in bestimmten Entwicklungsschritten (z.B. Digital Mockup[1] und Physical Mockup[2]) zu integrieren. Bisher können solche anwendungsübergreifenden Anwendungsfälle durch die Vielzahl an meist unabhängig entwickelten

[1] Digital Mockup bezeichnet ein möglichst wirklichkeitsgetreues, computergeneriertes Versuchsmodell, das hauptsächlich verwendet wird, um einen Teil der sehr teuren, realen Produktprüfung durch Computersimulationen zu ersetzen [Wika].

[2] Physical Mockup bezeichnet ein reales Versuchsmodell, an dem Funktionsweisen getestet und Probleme aufgezeigt werden sollen [Wikb].

Systemen und Komponenten aber nur mit sehr hohem manuellen Aufwand realisiert werden. An diesem Punkt setzt nun das PROCEED-Framework an, das relevante Produktdaten in ein einheitliches Datenschema integriert.

## 2.2 Bestandteile von PROCEED

Das Ziel dieser Arbeit ist die Visualisierung und Navigation durch komplexe Produktdaten. Zum besseren Verständnis der Hintergrundthematik werden in diesem Abschnitt die wichtigsten Begriffe und Bestandteile des PROCEED-Frameworks erläutert.

### 2.2.1 Produktdatenapplikation

Um die verschiedenen Entwicklungsaufgaben erledigen zu können, werden eine Vielzahl von Anwendungen verwendet. Jede dieser Anwendungen besitzt ein konzeptuelles Schema, welches die Beschreibung der jeweiligen Anwendung basierend auf Konzepten und ihren Zusammenhängen darstellt. Dieses Schema resultiert aus Langzeiterfahrungen und bietet somit einen hohen Geschäftswert, weshalb die jeweiligen Entwickler einer vollkommenen Offenlegung ihrer Anwendung kritisch gegenüberstehen. Nichtsdestotrotz wird in der Praxis auf eine große Menge solcher Anwendungen zurückgegriffen, wodurch zahlreiche Elemente mit teilweise überlappenden, also redundanten Informationen entstehen. Außerdem enthält jedes System neben dokumentierten Informationen auch nur selten dokumentiertes implizites Wissen. Dieses Wissen beschränkt sich auf die Experten und Nutzer der jeweiligen Anwendung. Eine explizite Dokumentation von Wissen, welches für die Experten und Mitarbeiter selbstverständlich ist, bringt diesem Teil der Anwender keinen Mehrwert, aber einen erheblichen Mehraufwand. Neben dem Geschäftswert ist vor allem das der Grund für die oftmals fehlende Dokumentation des impliziten Wissens. Für das allgemeine Verständnis aller anderen Nutzer kann das implizite Wissen jedoch wichtig sein. In einem größeren Zusammenhang mit anderen Applikationen kann sogar ein Mehrwert für alle Beteiligten entstehen. Ein Beispiel für implizites Wissen: Eine Komponente, z.B. „Scheinwerfer“, wird in dem System nur einmal dokumentiert (um Redundanzen zu vermeiden), aber im Fahrzeug zweimal benötigt. Erst beim Exportieren dieser Produktdaten wird der Scheinwerfer

dupliziert und ein linker und ein rechter angelegt. Deshalb ist es wichtig, auch das implizite Wissen zu berücksichtigen. Zum besseren Verständnis veranschaulicht Abbildung 2.1 die gesamte Problemstellung: Verschiedene Nutzer dokumentieren unterschiedliche Aspekte in unterschiedlichen Applikationen. Da diese unterschiedliche Anforderungen haben, sind die Datenmodelle heterogen statt homogen. Außerdem kommt hinzu, dass „gleiche“ Realweltobjekte unterschiedlich bezeichnet werden und somit eine direkte Zuordnung nicht immer automatisch möglich ist. Aus diesen Gründen müssen die Produktdaten in ein gemeinsames, konzeptionelles Schema (dieses wird im weiteren Verlauf Common Integration Ontology genannt, vgl. Abschnitt 2.2.4) integriert werden.

### 2.2.2 Produktdatenebenen

Grundsätzlich wird bei den Produktdaten nach der Transformation in eine Local Ontology (siehe Abschnitt 2.2.3) zwischen *Schema-Concepts* und *Individuals*[3] und innerhalb dieser Trennung zwischen den vier Produktdatenebenen *Product Data Collection*[4], *Object*, *Variant* und *Version* unterschieden. Beispiele für Schema-Concepts auf Objektebene sind z.B. Komponenten, Systeme, Netzwerke, Anforderungen und Tests. Jedes dieser Concepts hat eine gewisse Anzahl an Individuals. Außerdem können Objekte in unterschiedlichen Varianten vorhanden sein (z.B. ein Motorsteuergerät ist ein Objekt und kann für unterschiedliche Motorausführungen entwickelt werden, z.B. Diesel und Benziner). Da Produkte während der Entwicklung kontinuierlich weiterentwickelt werden, entstehen eine Vielzahl an Produktdatenversionen.

### 2.2.3 Local Ontologies

Da Produktdaten in jedem System individuell gespeichert werden (vgl. Abschnitt 2.2.1), müssen Vorkehrungen getroffen werden, bevor sie in die Common Integration Ontology integriert werden können. Damit ist vor allem eine Vereinheitlichung der Struktur aller Produktdaten (vgl. Abschnitt 2.2.2) durch eine vorherige Transformation in eine sogenannte *Local Ontology (LO)* gemeint. Diese Transformation ist wichtig, um die Datenmodellheterogenität zu

[3] Ausprägungen, vergleichbar mit Datenbankeinträgen

[4] Objekte werden mittels Product Data Collections gebündelt. Hierbei kann es sich z.B. um eine Fahrzeugbaureihe handeln.

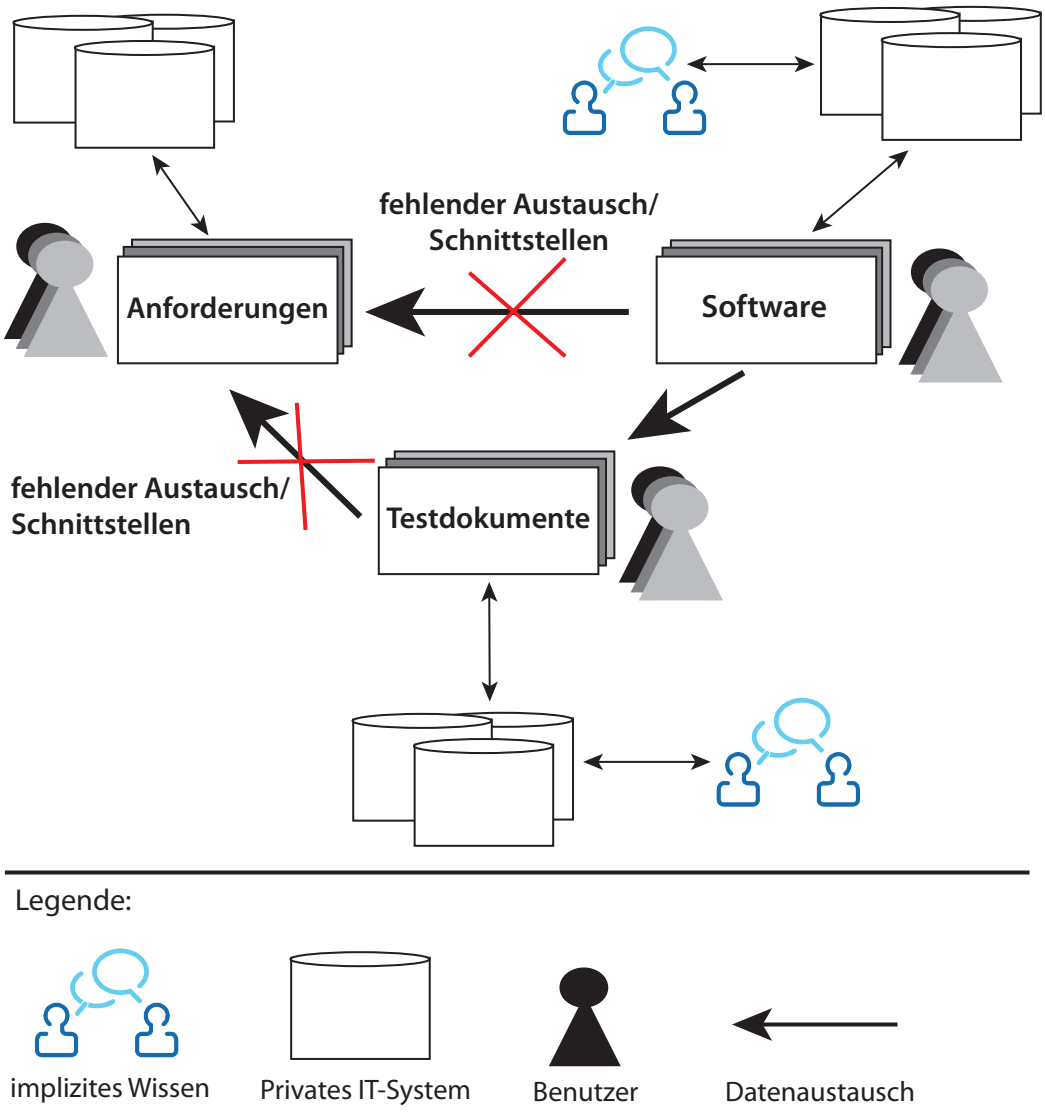


Abbildung 2.1: Verschiedene Nutzer dokumentieren unterschiedliche Aspekte in unterschiedlichen Applikationen. Da diese unterschiedliche Anforderungen haben, sind die Datenmodelle heterogen. Zusätzlich werden „gleiche“ Realweltobjekte unterschiedlich bezeichnet, wodurch eine direkte Zuordnung nicht immer automatisch möglich ist. Deshalb müssen die Produktdaten in die CIO integriert werden.

überwinden. Wie ein privates IT System seine Daten auf eine Local Ontology abbildet, bleibt den Verantwortlichen des Systems letztlich selbst überlassen. Die Transformation wiederum muss für alle Systeme durchgeführt werden. Um die unterschiedlichen Produktdatenebenen und den Zusammenhang von Schema-Concepts und deren Individuals zu veranschaulichen, zeigt die Abbildung 2.2 ein Beispiel für eine Local Ontology.

### 2.2.4 Common Integration Ontology

Die Local Ontologies sorgen für einen plattformunabhängigen und einheitlichen Zugriff auf Produktdaten aus unterschiedlichen IT Systemen. Die eigentliche Integration, also Zusammenführung von Produktdaten, findet aber in der *Common Integration Ontology (CIO)* statt. Genauer gesagt werden semantisch verwandte Elemente verschiedener Local Ontologies zu neuen Elementen zusammengefasst, die dann Teil der CIO sind. Diese integrierten Elemente bilden die Basis zur Abfrage von E/E-Produktdaten, weshalb diese weitere Abstraktionsschicht mitsamt seinen Elementen den Hauptfokus der weiteren Erkenntnisse und Entwicklungen dieser Arbeit bildet. Wegen möglichen Unterschieden in den einzelnen Local Ontologies besteht die Gefahr, dass Elemente aus mehreren dieser Ontologies das gleiche repräsentieren, aber in der Common Integration Ontology nicht zugeordnet werden können. Deshalb werden Abbildungsegeln benötigt, um trotzdem eine Zuordnung umsetzen zu können. Solche Regeln werden im Folgenden auch als *Mappings* bezeichnet. Da die Anzahl der Schema-Concepts überschaubar ist, kann ein Mapping auf dieser Ebene noch manuell durch einen Mitarbeiter vollzogen werden. Dabei müssen für ein Element aus einer Local Ontology und dessen passenden Elements aus der CIO Eigenschaften für das Mapping angegeben werden. Zum einen muss angegeben werden, in welcher Art die beiden Elemente in Beziehung stehen, also ob sie sich z.B. gleichen, das Element der Local Ontology nur einen Teil des anderen Elements darstellt oder umgekehrt. Zum anderen besitzt eine Mapping-Beziehung eine oder mehrere Regeln. Beispielsweise ähneln sich Elemente oftmals in ihrer Bezeichnung, was durch eine einfache Zeichenketten-Übereinstimmung bestimmt werden kann. Andere Möglichkeiten wären eine Zuordnung über reguläre Ausdrücke oder einer Zuordnung über eine Tabelle. Ist ein solches manuelles Concept-Mapping abgeschlossen, können daraus automatisch Mappings der zugehörigen Individuals abgeleitet und durchgeführt werden. Es kann jedoch vorkommen, dass sich keine passenden Mappings zwischen Individuals einer Local Ontology und einer CIO bilden lassen. Außerdem

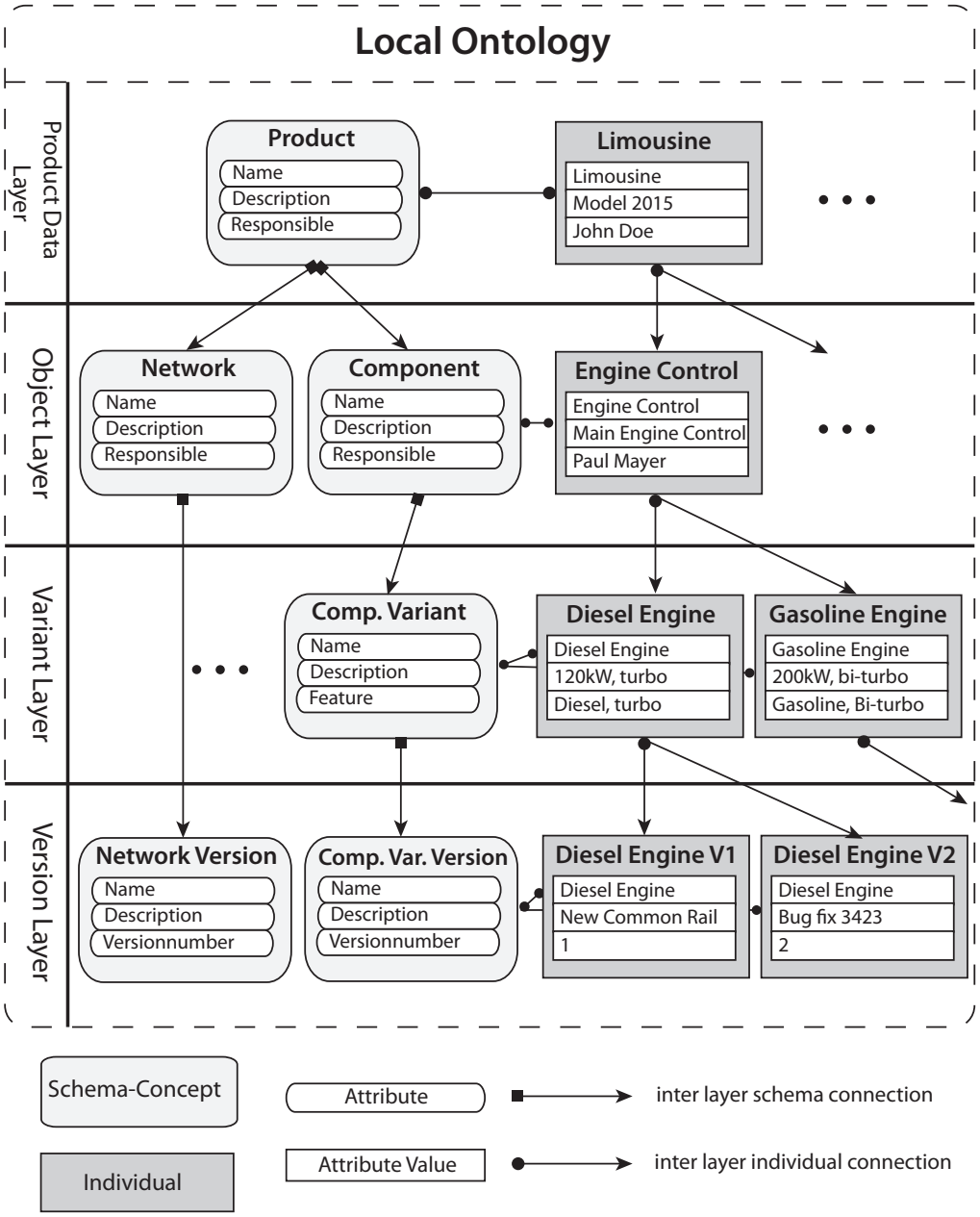


Abbildung 2.2: Beispiel einer Local Ontology mit allen dazugehörigen Produktdatenebenen und beispielhaften Schema-Concepts und Individuals.

kann es passieren, dass die Integration nicht vollständig automatisch geschehen ist und einige Abbildungen fehlen oder falsch vom Algorithmus erkannt worden sind. In diesem Fall müssen Dokumentaristen (*Data Quality Engineers*) diese Elemente manuell überprüfen und gegebenenfalls Mappings von Hand erstellen. Sind all diese Schritte vollzogen, findet man eine einheitliche Datenbasis vor, in der man alle notwendigen Schemata und Individuals samt Attributen vorfindet. Anhand dieser Basis können nun Datenabfragen, die einen Ausschnitt aus der Common Integration Ontology darstellen, für verschiedene Prüfungen und fachliche Anfragen durchgeführt werden. Da das Thema Mappings beziehungsweise Korrespondenzen zwischen den Produktdaten für das weitere Verständnis dieser Arbeit wichtig sind, dient die folgende Zusammenfassung samt Abbildung 2.3 nochmals als Hilfe.

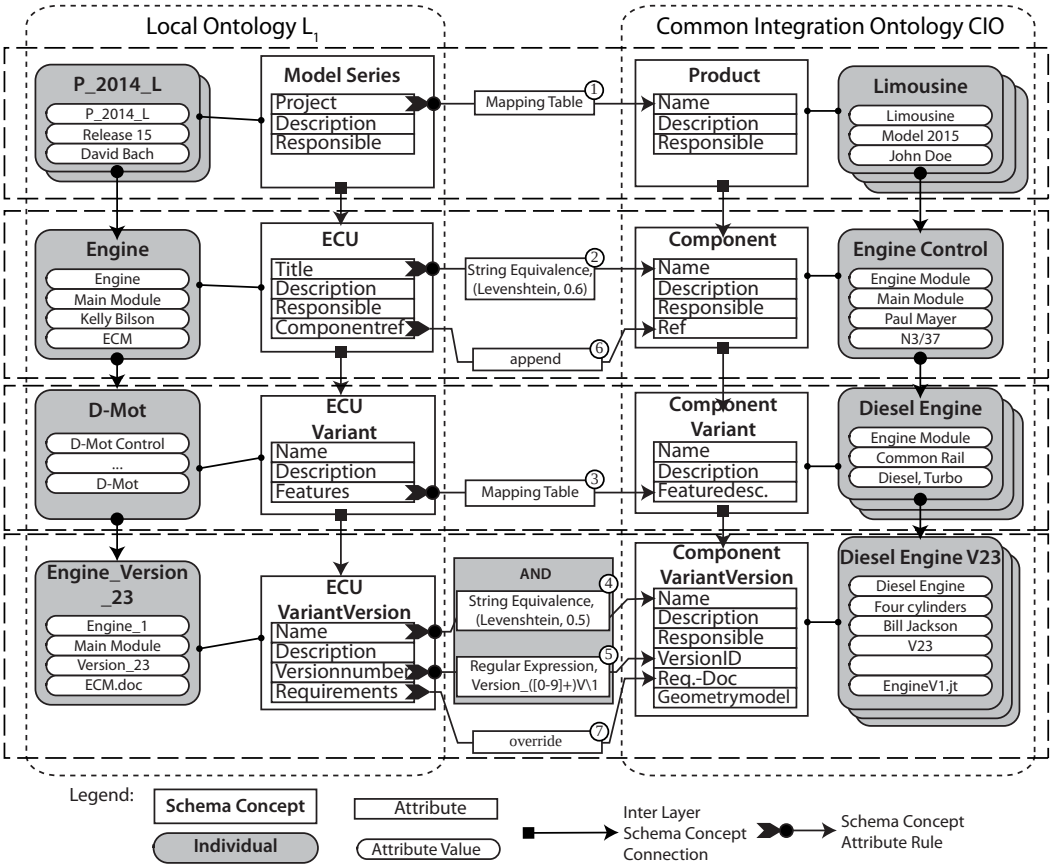


Abbildung 2.3: Beispiel von LO und CIO mit eingezeichneten Korrespondenzen

Beim Thema Mapping muss zwischen *Regeln* und *Aktionen* unterschieden werden: *Schema Concept Attribute Rules (SCARs)* beschreiben Regeln, um gleiche Realweltobjekte zu finden. Zeichenketten-Äquivalenzen (string equivalences) oder reguläre Ausdrücke (regular expressions) (vgl. Abbildung 2.3, Mittelspalte) sind Beispiele für solche Regeln, um zusammengehörige Elemente zu finden. Dabei kann festgelegt werden, dass mehrere Regeln zutreffen müssen um die Zuammengehörigkeit definitiv zu bestimmen. Noch detaillierter können solche Regeln mit Hilfe von AND- und OR-Operatoren definiert werden. Wenn eine Regel zutrifft, werden Attributewerte von Schema-Concepts einer Local Ontology in Attribute von Schema-Concepts der CIO kopiert. Dazu werden Aktionen definiert, die *Schema Concept Attribute Actions (SCAA)* heißen. In Abbildung 2.3 wird zum Beispiel durch Zeichenketten-Äquivalenzen festgestellt, dass das Element *Engine* der Local Ontology *L1* und das Element *Engine Control* aufgrund ihrer ähnlichen Namen zusammengehören. Somit können für diese beiden Elemente Aktionen definiert werden, um die Attributwerte von *Engine* in das Element *Engine Control* in der CIO zu kopieren.

## 2.3 Visualisierung und Navigation von Datenabfragen

Abfragen der Common Integration Ontology können bisher textuell oder grafisch erfolgen. Diese werden im PROCEED-Framework über einen Dialog erstellt und dann auf die CIO angewendet. Damit ist die eine Seite einer Datenabfrage schon im Framework verwirklicht worden. Sowohl die Darstellung von Abfrageergebnissen als auch die Navigation in diesen Ergebnissen sind bisher nicht berücksichtigt worden und sollen in dieser Arbeit näher untersucht werden. Wesentliche Gesichtspunkte, die dabei zu beachten sind, sind zum Beispiel die Frage, ob die Ergebnisse textuell dargestellt werden sollen oder ob man sich an einer grafischen Lösung orientiert. Eine weitere Frage ist, was überhaupt dargestellt werden soll, also ob man als visuelles Ergebnis lediglich einen Teilausschnitt einer Common Integration Ontology vorfindet oder passend zu den einzelnen verknüpften Elementen die Local Ontologies mitgeliefert werden. Wenn dies der Fall ist, gilt es zu untersuchen, was geschieht, wenn es zu einem Element der CIO zu viele Local Ontologies gibt, die noch sinnvoll dargestellt werden können. Sobald man dieses Problem gelöst hat, ergeben sich daraus jedoch direkt die nächsten Fragen: Wie kann man an dieser Stelle eine Navigation durch ein Ergebnis steuern, welche Möglichkeiten gibt es dabei und an welche Grenzen

stößt man? Als Rahmen um die ganze Entwicklung ist die Gestaltung im Sinne des Usability Engineerings gedacht, d.h. die Gestaltung fokussiert sich durchgehend auf die beteiligten Benutzern und wie sie ihre Aufgaben am effektivsten, effizientesten und mit einem gewissen Grad an Zufriedenheit erledigen können. Unter Berücksichtigung dieser Ziele soll dabei am Ende ein System entstehen, welches nicht nur ein ästhetisches Design besitzt, sondern auch den Aufgaben entsprechend gut umgesetzt worden ist. Einen tieferen Einblick in den Bereich der Usability gibt das Kapitel 3, in dem unter anderem der Begriff genauer erklärt und wichtige Elemente davon genannt und nähergebracht werden.

# 3 Grundlagen des Usability Engineerings

In dieser Arbeit werden Ansätze entwickelt in denen Teile der Common Integration Ontology als Ergebnisse einer Datenabfrage visuell dargestellt werden. Dabei wäre es auch möglich, die Ergebnisse in einer Tabelle oder einer anderen textuellen Beschreibung darzustellen. Bilder ermöglichen jedoch eine schnellere Entstehung eines mentalen Modells im Kopf, mit dem die wichtigsten Punkte eines Sachverhalts im Gehirn verstanden werden können. Bei Texten dauert dieser Vorgang länger und belastet meist das Arbeitsgedächtnis des Menschen in höherem Ausmaß [Seu13a]. Bilder haben folglich eine gewisse Mächtigkeit und einen Einfluss auf das medienspezifische Lernverhalten von Menschen. Deshalb beschäftigt sich dieses Kapitel zuerst mit der Bedeutung von Bildern und daraufhin mit dem Begriff Usability, der für die Entwicklung visueller Ansätze und Oberflächen von großer Bedeutung ist.

## 3.1 Bedeutungen und Funktionen eines Bildes

Bilder sowie Texte gehören zum Oberbegriff Multimedia. Dieser besteht aus einer technischen, einer sensorischen und einer semiotischen Ebene. Die semiotische Ebene ist die Ebene der Zeichenarten und behandelt Formen einer Darstellung wie Texte, Bilder und Diagramme. Vor allem jedoch verdeutlicht sie den Unterschied zwischen Text und Bild. Es gibt zwei Arten, um beispielsweise ein Auto zu beschreiben. Ein Auto kann man mit Hilfe der Symbolstrukturen A, U, T und O beschreiben. Eine andere Möglichkeit wäre die Verwendung von analogen Strukturen, wodurch ein Bild beziehungsweise eine grafische Darstellung eines Autos als Beschreibung entsteht. Der Vorteil von solchen sogenannten Depiktionen ist die Ähnlichkeit zum Begriff Auto, denn das Bild repräsentiert den eigentlichen Sachverhalt deutlich anschaulicher als die vier Buchstaben des Wortes Auto. Aufgrund dieser Tatsache ergeben sich einige Funktionen von Bildern (vgl. [Seu13b]):

- Bilder helfen dem Verständnis, denn sie helfen eine konkrete Vorstellung zu entwickeln und Prozesse vor dem geistigen Auge ablaufen zu lassen.
- Sie veranschaulichen Informationen aus Statistiken oder Tabellen, sodass man alles auf einen Blick sehen kann.
- Bilder können dynamisch sein und dadurch reale Bewegungen mental vorstellbar machen.
- Bilder helfen Probleme (z.B. häufige Stellen von Blechschäden an Autos) zu lösen, da eine bildhafte Beschreibung der Stellen einfach und offensichtlich ist.
- Bilder leiten konkret bei Handlungen an und helfen beim Behalten und Verstehen von Strukturen und Sachverhalten.

## 3.2 Medienspezifische Lerntheorien

Vergleiche von Text und Bild sowie die Erkenntnis einer unterschiedlichen menschlichen Auffassungsgabe sind durch verschiedene medienspezifische Lerntheorien wissenschaftlich untersucht und belegt worden. Die Grundtheorie, auf die all diese Theorien basieren, ist das Modell der dualen Kodierung von Paivio [Pai86]. Hier wird davon ausgegangen, dass Text und Bild verschiedene Systeme im menschlichen Gehirn ansprechen. Während abstrakte Worte lediglich das verbale System aktivieren, können nur konkrete, anschauliche Worte sowohl das verbale als auch das imaginale System aktivieren. Bilder jedoch aktivieren grundsätzlich beide menschliche Systeme. Durch diese duale Kodierung beziehungsweise der doppelten Abspeicherung im Gehirn kann man sich besser an Bilder als an Begriffe und Texte erinnern. Zu jedem Bild lässt sich also eine Beschreibung erstellen, umgekehrt jedoch nicht. Auch weitere Theorien wie die Cognitive Theory of Multimedia Learning [May01] belegen eine unterschiedliche Verarbeitung von bildhaftem und textuellem Inhalt und untermauern die Vorteile einer grafischen Darstellung. Ein weiteres Beispiel liefert das Modell des integrierten Text- und Bildverstehens [SB99]. In diesem Modell (vgl. Abbildung 3.1) werden die einzelnen Verarbeitungsschritte für Text und Bild dargestellt, wobei zwischen einer externen Darstellung, einer Oberflächenrepräsentation und einer tiefen Verarbeitung des Textes beziehungsweise Bildes (von unten nach oben) unterschieden wird.

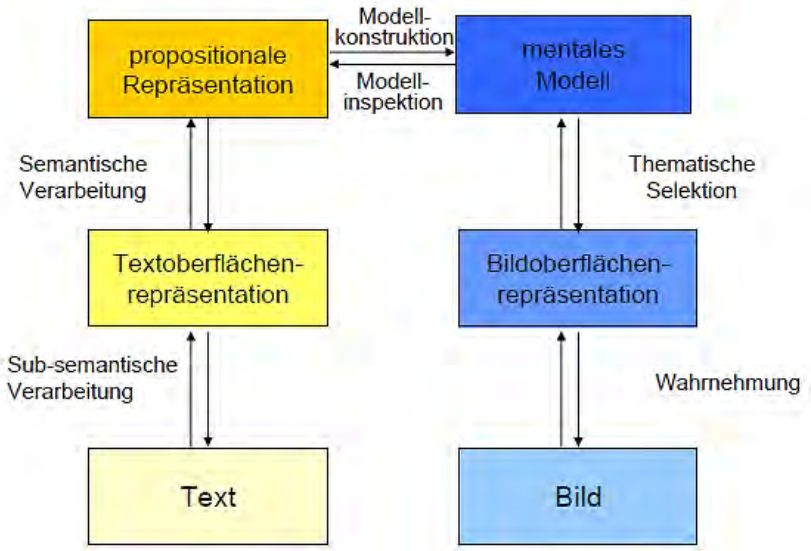


Abbildung 3.1: Modell des integrierten Text- und Bildverstehens, siehe [Seu13a].

Angewandt auf das PROCEED-Framework bedeutet dies, dass eine textuelle Beschreibung von Objekten eines Schema-Concepts beispielsweise zunächst auf subsemantischer Ebene verarbeitet wird, d.h. man merkt sich lediglich die Textoberflächenrepräsentation (hier: die Namen plus evtl. Attribute der Objekte und korrespondierenden Elemente). Basierend auf diesem Schritt folgt die semantische Verarbeitung, indem versucht wird, einzelne Begriffe, die die Kernaussage des Textes darstellen (Propositionen), aus dem Text zu bilden. Erst nach diesen beiden Schritten auf sprachlicher Ebene folgt bei einer textuellen Darstellung die Konstruktion eines bildhaften mentalen Modells. Aus den einzelnen abgeleiteten Begriffen wird hierbei eine analoge Repräsentation geschaffen, die letztlich der des Abfrageergebnisses ähnelt. Als Ergebnis besitzt man somit letztlich ein Bild im Kopf, das die einzelnen betroffenen Objekte der Datenabfrage beschreibt, welche entsprechend mit ihren jeweiligen korrespondierenden Elementen verknüpft sind. Bei einer bildhaften Repräsentation einer Struktur geschieht die Konstruktion eines mentalen Modells in dieser Theorie schneller. Das Bild muss zwar ebenfalls gedanklich auf seine Oberflächenrepräsentation reduziert werden, daraus kann jedoch bereits ein mentales Modell erstellt werden. Dies ist zwar

ein wesentlich informationsärmeres Bild, ermöglicht aber eine klare Vorstellung von dem gezeigten externen Bild.

Beim Textverstehen ist folglich ein interner Übersetzungsprozess beziehungsweise Sprachwechsel von Sprachzeichen zu Bildzeichen notwendig. Bei einem Bild entfällt dieser Schritt, man erreicht also schneller den gewünschten Zielpunkt des Verstehensprozesses. Somit kann für die visuellen Ansätze abgeleitet werden, dass es wesentlich sinnvoller ist, vor allem die Zusammenhänge zwischen Ergebniselementen und korrespondierenden Elementen bildhaft statt textuell darzustellen. Dadurch entsteht beim Benutzer ein schnelleres mentales Modell und er versteht leichter, welche Elemente mit welchen in Verbindung stehen.

## 3.3 Grundlagen der Bildgestaltung aus psychologischer Sicht

Aufgrund der Erkenntnisse des vorherigen Abschnitts sind Bilder beziehungsweise grafische Darstellungen besser als textuelle Beschreibungen zur Veranschaulichung von Sachverhalten geeignet. Dieser Abschnitt erläutert nun, was daraus für die Nutzung beziehungsweise Gestaltung von Bildern abgeleitet werden kann. Daraufhin wird auf die Auswirkung auf das Arbeitsgedächtnis eingegangen.

### 3.3.1 Nutzung eines Bildes

Betrachtet man die in Abschnitt 3.1 genannten Funktionen eines Bildes, so lassen sich daraus zwei wesentliche Punkte bezüglich der Nutzung eines Bildes zusammen mit dem Modell des integrierten Text- und Bildverstehens ableiten. Erstens müssen Bilder sowohl das Verstehen als auch die Wahrnehmung fördern. Zweitens gehören zum Verständnis eine klare Bildaussage sowie eine pragmatisch einfache Darstellung. Wichtig ist hierbei zum Beispiel die Wahl eines passenden Diagramms und der farblichen Betonung der visuellen Informationen. Möchte man beispielsweise Vergleiche mit Hilfe eines Diagramms darstellen, sollte man Flächen oder Volumen vermeiden, da hierbei Unterschiede unterschätzt werden können, die durch ein Balkendiagramm viel deutlicher ausfallen würden. Außerdem sollte

man sich bei der Darstellung auf das Wesentliche konzentrieren, statt Details einfließen zu lassen. Details sind meist nicht für das eigentliche Verständnis des Bildes notwendig, sondern hindern den Betrachter eher in seiner Wahrnehmung.

### 3.3.2 Einschränkungen durch die Arbeitsgedächtniskapazität

Bezüglich der Wahrnehmungsförderung ist bei der Gestaltung neben der Beachtung von Gestaltgesetzen (z.B. Gesetz der Nähe, der Ähnlichkeit oder der Geschlossenheit) auch die beschränkte Kapazität des menschlichen Arbeitsgedächtnisses zu berücksichtigen. Informationen können in unendlichen Mengen vom Arbeitsgedächtnis aufgenommen werden, aber nur wenige davon direkt verarbeitet werden. Allgemein bekannt ist die grobe Schätzung von ca. 7 Einheiten, die das menschliche Arbeitsgedächtnis an Informationen verarbeiten kann [Seu13a]. Als Schlussfolgerung aus dieser Beschränkung muss für eine visuelle Darstellung gelten, dass ihr Inhalt in diesen Rahmen passt. Ohne Beachtung dieser Grenze kann es zu einer von Sweller und Chandler erforschten kognitiven Überlastung kommen. In ihrer Cognitive Load Theory [CS91] beschreiben sie drei verschiedene Belastungsarten des Arbeitsgedächtnisses (vgl. Abbildung 3.2).

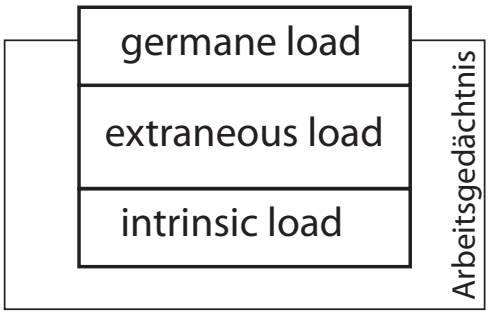


Abbildung 3.2: Der intrinsic load, der extraneous load und der germane load haben Einfluss auf die Arbeitsgedächtniskapazität, vgl. [Seu13c]

Dabei gelten der sogenannte *intrinsic load* und der *extraneous load* zu den beiden Loads, die man so gering wie möglich halten sollte. Ersterer bezieht sich auf die Schwierigkeit beziehungsweise die Komplexität des zugrundeliegenden Inhalts. Je komplexer das Thema, desto mehr Aufwand benötigt das Arbeitsgedächtnis, um die Informationen zu verarbeiten. Dieser Teil kann lediglich durch das Vorwissen des Betrachters verbessert werden, wodurch der Inhalt verständlicher wird. Das größte Problem der Verarbeitung stellt jedoch der *extraneous*

*load* dar. Je unstrukturierter und unübersichtlicher eine Darstellung gestaltet ist, desto höher ist diese Art der kognitiven Belastung. Deshalb kommt es bei der Gestaltung von visuellen Informationen vor allem darauf an, durch eine gute Darstellungsweise die Belastung zu reduzieren. Die restliche Kapazität des Arbeitsgedächtnisses wird durch den sogenannten *germane load* aufgefüllt. Diese Belastung ist die einzige der drei genannten, die es zu fördern gilt. Mit Hilfe dieser positiven Belastung wird der Lernprozess in Gang gesetzt, der dem Betrachter hilft, die gezeigten Strukturen und Zusammenhänge zu verstehen und in seinem Gehirn abzuspeichern. Je mehr man diese drei Belastungen des menschlichen Arbeitsgedächtnisses bei der Gestaltung von visuellen Darstellungen berücksichtigt, desto eher kann die mitgelieferte Information verarbeitet und verstanden werden [Seu13c].

## 3.4 Einschränkungen des Begriffs „visueller Ansatz“

Nachdem die vorherigen Kapitel die Eigenschaften und Verwendung von Bildern erläutert haben, muss die Verwendung von bildhaften Darstellungen relativiert und eingeschränkt werden. Hinsichtlich dieser Arbeit wurde sich zwar für eine visuelle statt für eine schlichte, tabellenähnliche Darstellung entschieden, was jedoch nicht bedeutet, dass jedes einzelne Element der Produktdaten im System durch einen analogen Stellvertreter repräsentiert wird. Dies würde einen erheblichen Lernaufwand für die betroffenen Mitarbeiter bedeuten, da diese sich alle Bilder der sehr hohen Vielzahl an E/E-Produktdaten vor ihrer eigentlichen Arbeit einprägen müssten. Somit werden für die einzelnen Produktdaten symbolische Strukturen verwendet. Dies erleichtert das Verständnis ausreichend genug, um dem Benutzer anzuzeigen, um welche Produktdaten es sich handelt. Mit „visueller Ansatz“ wird in dieser Arbeit mehr das Gesamtbild verstanden, welches von den Produktdaten durch die einzelnen Datenabfragen geschaffen wird. Welche Gesamtdarstellungen das sein können, wird in Kapitel 4 analysiert.

## 3.5 Usability Engineering

Die Entwicklung einer visuellen Darstellung von Informationen bringt trotz der in Abschnitt 3.3 erläuterten Vorteile und Funktionen auch, wie im vorherigen Abschnitt erwähnt, einige

Probleme mit sich. Diese darf man keinesfalls unterschätzen, denn sie sind im Endeffekt dafür verantwortlich, ob die Mitarbeiter später mit dem System umgehen können. Sind Zusammenhänge in der Gesamtdarstellung nicht klar ersichtlich oder Arbeitsschritte nur schwer durchführbar, so erschwert ein solches interaktives System die Arbeit. Um dies zu verhindern, wird heutzutage für viele Softwareentwicklungen oftmals auf das Konzept des Usability Engineerings zurückgegriffen. Mit Hilfe des Usability Engineerings lassen sich Faktoren bestimmen, die Einfluss auf die Benutzbarkeit eines Systems haben. Gleichzeitig bietet es Möglichkeiten, diese Faktoren von der ersten bis zur letzten Entwicklungsphase zu berücksichtigen. Doch bevor diese Punkte ausführlich erläutert werden, wird zunächst der Begriff *Usability* näher betrachtet.

Für den Begriff *Usability* gibt es vielerlei Möglichkeiten für eine Übersetzung. Dabei sind die meisten deutschen Wörter keine passenden Umschreibungen. In [RF13] kam man zu dem Schluss, dass sowohl „Gebrauchstauglichkeit“ als auch „Benutzerfreundlichkeit“ nicht das aussagen, was man mit *Usability* im eigentlichen Sinne meint. Die „Benutzbarkeit“ eines interaktiven Systems war dabei noch die für die Autoren am ehesten geeignete Übersetzung. Dabei unterscheidet man eine Beschreibung des Begriffs *Usability* in einem engeren und einem weiteren Sinne. Grundsätzlich dient es als Gütekriterium für die Gestaltung einer Benutzeroberfläche, doch diese sollte auch leicht und verständlich verwendet werden können. Somit bedeutet eine hohe *Usability*, dass vorgesehene Benutzer das System einfach bedienen und ihre Aufgaben effizient, effektiv und mit einem gewissen Maß an Zufriedenheit bewerkstelligen können. [RF13] verdeutlicht diese anhand eines guten Beispiels: „Die Usability eines Hammers zum Einschlagen von Nägeln kann gut sein. Doch sie wird ziemlich schlecht ausfallen, wenn ihre Aufgabe darin besteht, Schrauben einzudrehen“. Für den Begriff *Usability* wurde sogar ein eigener ISO Standard (ISO-NORM 9241-11) definiert. Allgemein wurde für den Bereich Ergonomie der Mensch-System-Interaktion, der diesen und weitere verwandte Themen zur Gestaltung von Dialogsystemen aufgreift, die Normreihe 9241 definiert.

Das Ziel von Usability Engineering ist die „Konzeption und Einführung einer Methodik zur Etablierung und Verbesserung von User-Interface-Entwicklungsarbeiten als integraler Bestandteil von Software- und Systementwicklungsprozessen“ [Off13c]. Dies ermöglicht unter anderem eine frühzeitige und kontinuierliche Zusammenarbeit am Gesamtprozess der Systementwicklung von Auftraggeber und Auftragnehmer. Dadurch verbessern sich die User-Interface-Entwicklungsprozesse und die Qualität der Benutzeroberflächen steigt

automatisch an. Das Usability-Konzept ist praxisorientiert und wird somit dauerhaft durch Erfahrungen aus praktischen Projekten angereichert. Es basiert auf einem Prozessmodell mit definierten Phasen und Prozessschritten (siehe Abschnitt 3.9) und bietet durch Methoden und Hilfsmittel den gewünschten Mehrwert für die Entwicklung von interaktiven Systemen. Außerdem bietet das Vorgehen eine frühe Verfügbarkeit von Oberflächen, wodurch sich sowohl Entwickler als auch Benutzer frühzeitig näher mit dem System identifizieren können. Hauptsächlicher Profiteur der Berücksichtigung der Usability im Projekt ist der Auftraggeber. Handelt man nach den vorgegebenen Phasen und Prozessschritten, ist sichergestellt, dass das System bei Auslieferung den Erwartungen entspricht. Das System ist damit automatisch praxistauglich und die Mitarbeiter müssen lediglich noch eingearbeitet werden.

Dass ein solches Vorgehen seit Langem sinnvoll und notwendig ist, zeigt der schon 1994 veröffentlichte *CHAOS Report* der Standish Group [The94], der erschreckende Tatsachen der „aktuellen“ Softwareentwicklung offenlegte. Laut diesem Bericht werden 31% aller Softwareprojekte vor ihrer Fertigstellung abgebrochen. Von denjenigen Projekten, die vollendet werden, kosten 53% davon fast doppelt so viel wie vorher berechnet. Lediglich 16% aller Projekte werden zeit- und budgetnah fertiggestellt. Als Hauptursachen dafür wurden eine fehlende Einbindung der späteren Benutzer und unklare Aussagen über Anforderungen ausgemacht.

Bei der Entwicklung eines interaktiven Systems ist also unbedingt notwendig, die Benutzer und deren Aufgaben in diesem System zu berücksichtigen. Das Schalenmodell der ganzheitlichen Gestaltung, welches in Kapitel 3.8 beschrieben wird, geht auf diesen Teil nochmals detaillierter ein. Vorab gibt es aber drei wichtige Fragen, über die man sich als Entwickler klar werden muss: Was kann man überhaupt gestalten? Nach welchen Grundsätzen und Kriterien kann man gestalten und wie sollte man bei der Gestaltung vorgehen?

## 3.6 Benutzerschnittstellen

Im Fokus des Usability Engineerings stehen Systeme, die eine Interaktion mit einem Benutzer erfordern. Dies bedeutet meistens automatisch, dass eine grafische Benutzeroberfläche (GUI) vorliegt, die dem Benutzer die Informationen darstellt. In Bezug auf das PROCEED-Framework ist es ebenfalls ein solch beschriebenes System, welches als Desktopanwendung

auf den Mitarbeiter-PCs aufgerufen wird. Das IFIP-Modell [Dzi83], welches früher als Grundlage der Standardisierung von Benutzerschnittstellen diente, beschreibt für solche Systeme drei anwendungsunabhängige Benutzerschnittstellen: Eine Ein-/Ausgabeschnittstelle, eine Dialogschnittstelle und eine Werkzeugschnittstelle. Diese drei Teile stehen technisch gesehen zwischen Benutzer und Computer und bilden diejenigen Bereiche ab, die man gestalten kann [Off13b].

### 3.6.1 Ein-/Ausgabeschnittstelle

Innerhalb der Ein-Ausgabeschnittstelle gibt es einen Dialogbereich, mit dessen Hilfe ein Benutzer verschiedene Funktionen (z.B. Bearbeiten, Speichern, Hilfe) in dem Programm ausführen kann. Meistens darunter befindet sich der eigentliche Ein- und Ausgabebereich der Oberfläche, in welchem der Ausschnitt der Common Integration Ontology und die korrespondierenden Elemente dargestellt sind. Statusinformationen über den Inhalt dieses Bereichs werden meist am unteren Rand angebracht [Off13b].

### 3.6.2 Dialogschnittstelle

Der oben genannte Dialogbereich stellt eine eigene Schnittstelle dar. Zu ihm gehören Untermenüs und Dialogboxen, mit denen Einstellungen getätigt werden können. Diese Einstellungen beziehen sich meist auf Veränderungen bezüglich des dargestellten Inhalts der Ein-/Ausgabeschnittstelle [Off13b]. Im PROCEED-Framework können dies beispielsweise Schaltflächen oder Funktionsmenüs für die Ergebniselemente sein, um beispielsweise ihre Attribute anzuzeigen oder zu verbergen.

### 3.6.3 Werkzeugschnittstelle

Die Werkzeugschnittstelle ist eher für umfangreichere Systeme (z.B. Betriebssysteme) gedacht, in denen man zusätzlich noch eine Desktopansicht mit verschiedenen Verknüpfungen vorfindet. Dazu gehören auch eine Startleiste und das Startmenü. Diese Schnittstelle kann im Rahmen dieser Arbeit jedoch vernachlässigt werden [Off13b].

## 3.7 Gestaltungsgrundsätze

Hat man diejenigen Teile des Systems identifiziert, die es zu gestalten gilt, sollte man sich daraufhin bei der Gestaltung an gewisse Grundsätze halten. Speziell hierfür wurde ebenfalls ein Standard (EN ISO 9241, Teil 110) eingeführt, der sieben Grundsätze für die Gestaltung und Bewertung von interaktiven Systemen (auch: Dialogsysteme) definiert. Wenn im Folgenden von Dialogen gesprochen wird, so meint man damit eine Interaktion zwischen einem Benutzer und einem Dialogsystem, um ein bestimmtes Ziel zu erreichen. Die sieben Prinzipen des ISO-Standards (vgl. [Off13b]) umfassen:

- **Aufgabenangemessenheit**: Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen. Zu diesem Punkt gehören die vier folgenden Anforderungen an ein Dialogsystem, um dieses Ziel zu erreichen. Die Primäraufgabe eines Dialogsystems ist dessen *Nützlichkeit*. Alle Funktionen und Informationen, die der Benutzer zu seiner Aufgabenerledigung benötigt, sollen ihm angeboten werden. Um zu verhindern, dass der Benutzer durch Handhabungsprobleme im Dialog belastet oder von seiner Aufgabenerledigung abgehalten wird, sollen die Informationen komfortabel angeboten werden (*Komfort*). Ein gewisses Maß an *Übersichtlichkeit* soll die Wahrnehmung, Lesbarkeit, Orientierung und Aufmerksamkeit steuern und unterstützen. All dies kann jedoch nur dann funktionieren, wenn das System überhaupt verfügbar ist (*Verfügbarkeit*). Ein System kann einerseits nur dann benutzt werden, wenn eine gewisse Stabilität vorhanden ist und ein Systemausfall verhindert werden kann. Andererseits wirkt sich auch das Antwortzeitverhalten auf die Verfügbarkeit aus.

- **Erwartungskonformität**: Ist ein Dialog konsistent und entspricht den Merkmalen des Benutzers (z.B. Kenntnisse, Ausbildung, Erfahrung) sowie anerkannten Styleguides und Gestaltungsregeln, nennt man ihn erwartungskonform. Dialoge sollten sich einheitlich verhalten, bei ähnlichen Aufgaben ähnlich aussehen und Rückmeldungen bezüglich des Systemzustands liefern.

- **Selbstbeschreibungsfähigkeit**: Jeder Dialogschritt sollte durch eine entsprechende Rückmeldung verständlich sein. Eine Beschreibung sollte spätestens auf Anfrage des Benutzers geliefert werden. Dadurch wird eine Orientierung innerhalb mehrerer

Schritte des Dialogsystems gewährleistet. Der Benutzer weiß damit immer, wo er sich befindet, was er tun kann, wie er dorthin kam und wohin er von dort aus gelangen kann.

- **Steuerbarkeit**: Ist der Benutzer in der Lage, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis sein Ziel erreicht ist, so nennt man einen Dialog steuerbar.

- **Individualisierbarkeit**: Da Benutzer verschiedene Fähigkeiten und Vorlieben besitzen, sollte ein Dialog individuelle Anpassungen an die Erfordernisse der Arbeitsaufgabe zulassen. Diese Anpassungen sollen jedoch nur in bestimmten Grenzen möglich sein (z.B. Anbieten alternativer Bedien- und Darstellungsformen, individuelle Bildschirmaufteilung, Farb- und Schriftwahl).

- **Lernförderlichkeit**: Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet. Dazu gehören die Unterstützung relevanter Lernstrategien, das Wiederauffrischen von Gelerntem und das Zugänglichmachen von Regeln und zugrundeliegenden Konzepten für den Benutzer.

- **Fehlertoleranz**: Trotz erkennbar fehlerhafter Eingaben soll das Arbeitsergebnis entweder mit keinem oder nur mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden können.

All die genannten Grundsätze stellen jedoch nur allgemeine Leitlinien dar, die man bei der Gestaltung und Bewertung von Systemen berücksichtigen soll. Dabei hängt die Art und Weise jeder einzelnen Umsetzung von den Arbeitsaufgaben, den Merkmalen des Benutzers und der Arbeitsumgebung ab. Der einzige und wichtigste Grundsatz an ein System ist die Aufgabenangemessenheit. Speziell meint man damit die Nützlichkeit und den Komfort eines Systems, um Arbeitsziele effektiv und effizient zu erreichen.

## 3.8 Allgemeines Vorgehen bei der Gestaltung

Auf die Frage, wie man bei der Gestaltung vorgehen sollte, helfen verschiedene Grundmodelle der Mensch-Maschine-Interaktion bei der Beantwortung. Einen ersten Anhaltspunkt

dafür liefert das ABC-Modell [ITW]. Dieses Modell umfasst die drei Sichten *Angemessenheit*, *Benutzer* und *Computer* und bildet „die Beziehungen zwischen der Aufgabe an sich, dem Benutzer und dem Computer“ ab. Ein weiteres Modell, das Schalenmodell der ganzheitlichen Gestaltung, greift unter anderem die Punkte dieses Modells auf und verdeutlicht vor allem die Notwendigkeit einer ganzheitlichen Sicht. Ganzheitlichkeit bedeutet dabei, dass zur Gestaltung und Bewertung von interaktiven Systemen nicht nur die Software und Hardware berücksichtigt werden sollen. Der Aufbau und Ablauf aller Arbeitsschritte, die von Benutzern durchgeführt werden müssen, sollen ebenso in die Entwicklung miteinfließen. Die Reihenfolge (in Abbildung 3.3 als Bewertungsrichtung bezeichnet) der Vorgehensweise ist dabei wie folgt vorgegeben (vgl. [Off13a]):

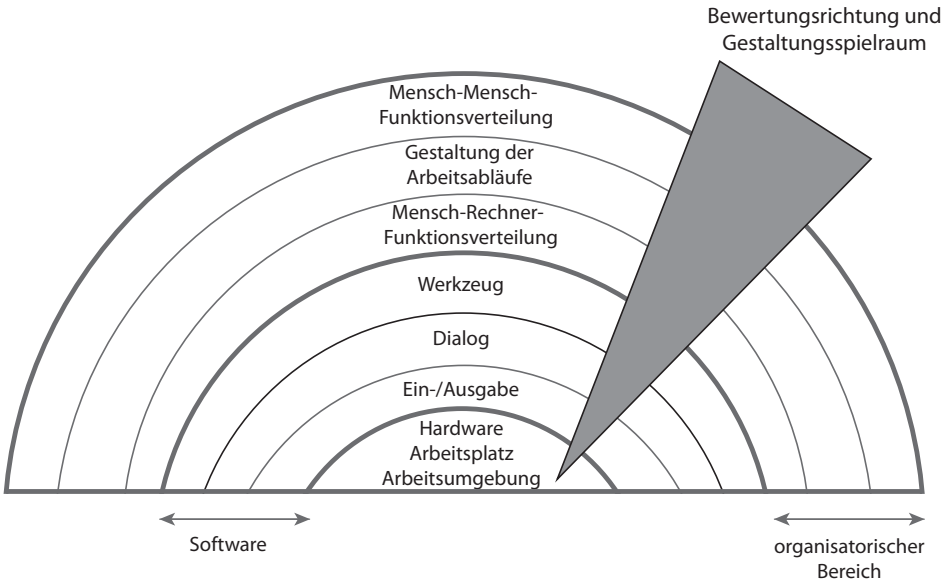


Abbildung 3.3: Das Schalenmodell der ganzheitlichen Gestaltung, vgl. [Off13a]

1. **Mensch-Mensch-Funktionsverteilung**: In diesem ersten Schritt konzentriert man sich einzig und allein auf die Aufgabenverteilung zwischen den betroffenen Benutzern. Meistens gibt es bestimmte Benutzerrollen, die verschiedene Aufgaben in ihren jeweiligen Abteilungen zu erledigen haben.

2. **Gestaltung der Arbeitsabläufe**: Nachdem alle Aufgaben unter den Mitarbeitern identifiziert wurden, muss geklärt werden, wie der Arbeitsablauf eines Benutzers einer jeden Benutzerrolle aussieht. Somit steht hier die Reihenfolge der Teilschritte einer Aufgabe im Vordergrund.

3. **Mensch-Rechner-Funktionsverteilung**: Die dritte Ebene in diesem Modell klärt die Frage, welche Aufgaben vom Benutzer selber getätigt werden sollen und welche vom Computer übernommen werden. Die Kommunikation mit anderen Abteilungen oder Kunden beispielsweise wird viel einfacher von einem Mitarbeiter durchgeführt. Die Darstellung von Daten oder komplexe Rechenleistungen sollten jedoch lieber einem PC überlassen werden.

4. **Software/Hardware**: Die eigentlich zu entwickelnde Software wird erst in einem letzten Schritt hinsichtlich der Gestaltung berücksichtigt. Dazu gehören die in Abschnitt 3.6 genannten Schnittstellen (Ein-/Ausgabe, Dialog, Werkzeuge).

Diese Reihenfolge, bei der man sich zuerst an den Aufgaben und den Benutzern orientiert, sorgt letztlich dafür, dass auf „die Aufgabenbedürfnisse und die Eigenschaften des Benutzers Rücksicht genommen worden ist“ [Off13a].
Beide genannten Modelle machen deutlich, dass es bei der Entwicklung eines interaktiven Systems mehr als die Software zu beachten gilt. Die Analyse der Benutzer und ihrer jeweiligen Aufgaben ist für ein System im Sinne einer hohen Benutzbarkeit unabdingbar. Auf diesen Erkenntnissen aufbauend stellt das folgende Kapitel das Referenzmodell für Usability Engineering der Daimler AG vor, anhand welchem die visuellen Ansätze dieser Arbeit umgesetzt werden.

## 3.9 Das Referenzmodell für Usability Engineering der Daimler AG

Der Mangel an Aufgaben- und Benutzerberücksichtigung stellt gleichzeitig die Motivation für den Einsatz von Usability Engineering dar. Geht man bei der Entwicklung einer Benutzeroberfläche nicht auf diese Punkte ein, hat das möglicherweise verheerende Auswirkungen auf

das gesamte System. Das Problem heutiger Systementwicklungsprozesse (vgl. Abbildung 3.4) ist jedoch, dass kaum Spielraum für Eingreifmöglichkeiten geboten ist.

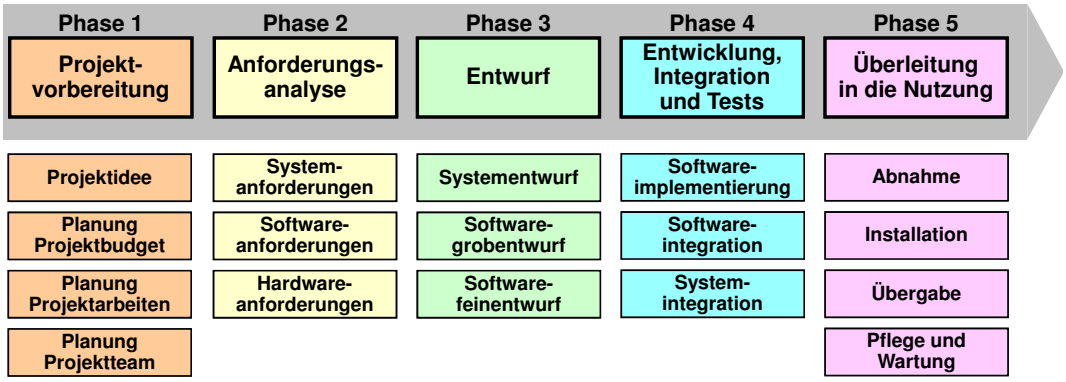


Abbildung 3.4: Typischer Systementwicklungsprozess, siehe [Off13c].

Zu Beginn findet die Projektvorbereitung statt, in der grundlegende Thematiken wie Budget, Teamaufteilung und Projektidee besprochen werden. Nach einer Analyse der System-, Software- und Hardwareanforderungen werden dazu passende Entwürfe entwickelt. Stehen ausgereifte Entwürfe zur Verfügung, werden diese implementiert und die Software sowie das System integriert. Ein letzter Schritt dient sowohl der Abnahme, Installation und Übergabe als auch der Pflege und Wartung des Systems. Da in keinem einzigen der genannten Schritte eine Evaluierung der Entwürfe mit den Benutzern stattfindet, kann erst ganz spät ein Statement über das System abgegeben werden. Dadurch entspricht das Resultat otmals einem gebrauchsuntauglichen Produkt. Weitere Hauptursachen dafür sind organisatorische Unzulänglichkeiten, eine saubere Dokumentation und zeitliche Verzögerungen aufgrund von nachträglichen einzuarbeitenden Funktionalitäten. Diese Probleme können grundsätzlich nur durch ein teures Redesign behoben werden. Das bedeutet, dass die Entwicklung unter Umständen bis zurück zur Anforderungsanalyse gehen muss, um die neuen Erkenntnisse einzuarbeiten. Der Einsatz des Referenzmodells für Usability Engineering der Daimler AG soll diesen Problemen von Beginn der Entwicklung an entgegenwirken. Statt auf eine Diagnose am Ende der Entwicklung, baut das Referenzmodell auf eine „Therapieorientierung mit ganzheitlichem Ansatz“ [Off13c]. Wie in Abbildung 3.5 dargestellt, erfolgt nach jeder Phase eine Evaluation, die in den meisten Fällen direkt mit den späteren Benutzern stattfindet.

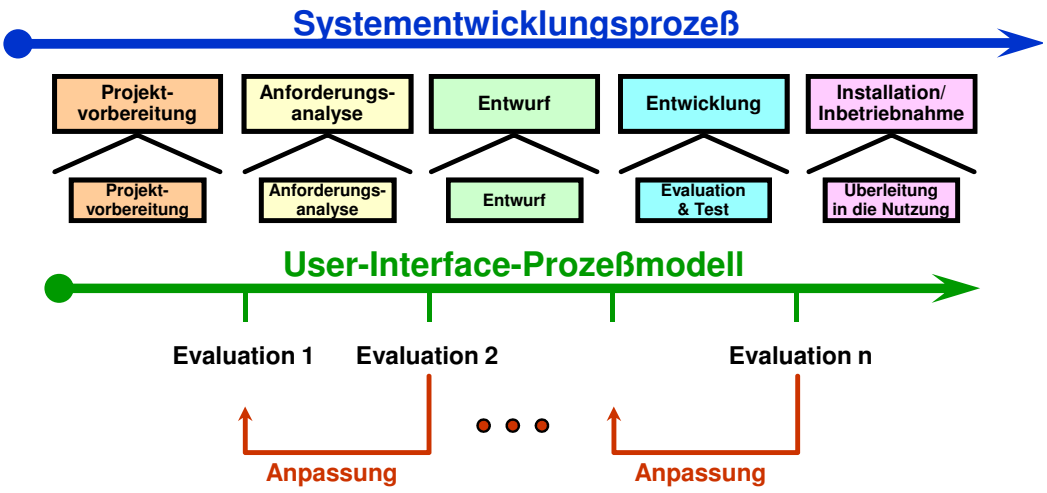


Abbildung 3.5: Therapieorienterter Entwicklungsansatz, siehe [Off13c].

Diese Art der Modellstruktur kann dazu führen, dass Phasen oder Prozesse einer Phase iterativ durchlaufen werden müssen, bis ein zufriedenstellendes Ergebnis erreicht ist. Durch diese schrittweise und iterative Begleitung des Software-Entwicklungsprozesses können Verbesserungen beziehungsweise Änderungen durch wenige kostengünstige Schritte (vor allem in den ersten Entwicklungsphasen) vollzogen werden.

Das Referenzmodell für Usability Engineering der Daimler AG (siehe Abbildung 3.6) definiert für alle Phasen eine Vielzahl an Prozessschritten und Einzelaktivitäten. Dabei gilt, dass nicht jeder einzelne Schritt für jedes Projekt notwendig ist. Ein geringes Projektbudget oder eine beschränkte Anzahl von zu entwickelnden Dialogen sind Beispiele für projektspezifische Randbedingungen, weshalb man auf manche Schritte verzichten kann. Es gibt jedoch auch Schritte, die unbedingt Teil der Entwicklung sein müssen, um die Vorteile des Usability Engineerings (vgl. Abschnitt 3.5) nutzen zu können. Zu diesen Schritten zählen die Benutzerprofilanalyse, kontextuelle Aufgabenanalyse und Usability-Tests aus den Phasen Anforderungsanalyse, User-Interface-Entwurf und Evaluationen und Tests [Off13c].

Um einen Überblick über die Phasen des Referenzmodells zu geben, werden die Hauptziele und -bestandteile jeder einzelnen im Folgenden kurz zusammengefasst:

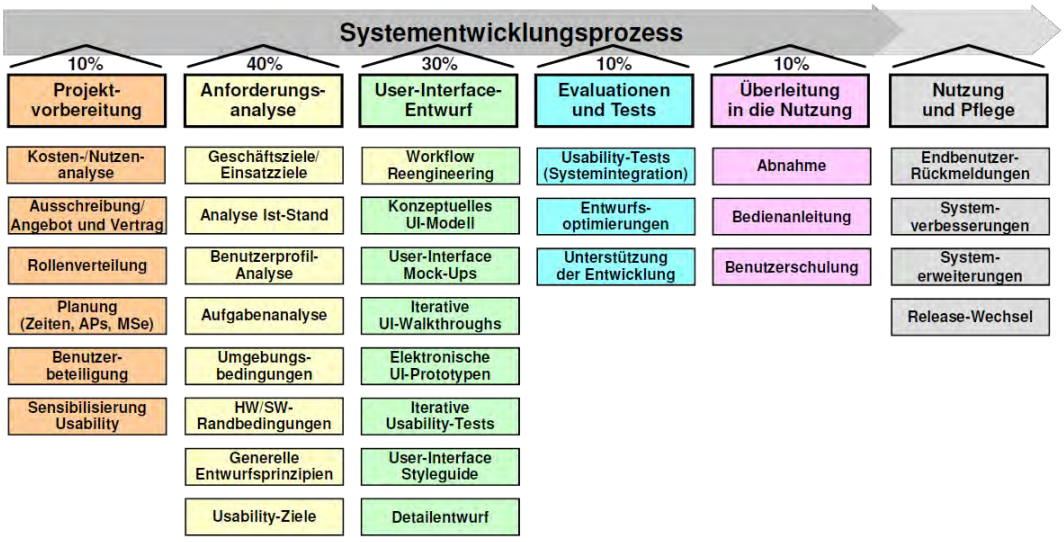


Abbildung 3.6: Das Referenzmodell der Daimler AG, siehe [Off13c].

1. **Projektvorbereitung**: Da eine frühzeitige Einplanung von Usability Engineering in den Softwareentwicklungsprozess für den Erfolg eines Projektes wichtig ist, werden schon in dieser ersten Phase alle erforderlichen Maßnahmen getroffen, um dies sinnvoll und reibungslos durchführen zu können. Dazu gehören hauptsächlich organisatorische Aufgaben wie die Erstellung von Angeboten oder Ausschreibungen, die Erstellung von Zeitplänen, Arbeitspaketen und Meilensteinen sowie einer Rollenverteilung aller an dem Projekt beteiligten Personen. Am Ende dieser Phase sollten alle Vorbereitungen vollzogen und die Mitarbeiter für das Thema Usability sensibilisiert sein [Off13d].

2. **Anforderungsanalyse**: Mit ca. 40% der Entwicklungszeit stellt die Phase der Anforderungsanalyse den aufwändigsten und wichtigsten Teil der Entwicklung dar. Durch sie werden wesentliche Basisarbeiten zur letztendlichen Gestaltung des Systems durchgeführt. Das Hauptziel dieser Phase ist, so viele Informationen wie möglich über den jetzigen Systemstand, die Benutzer und deren Aufgaben zu erhalten. Je besser die Analyse ausfällt, desto eher können passende Grundanforderungen an das zu entwickelnde System definiert werden. Zusätzlich werden hier Randbedingungen, die durch die Umgebung des Systems und der Hard- und Software anfallen, sowie gene-

relle Entwurfsprinzipen geklärt, die sich auf firmeninterne Gestaltungsregeln beziehen [Off13e].

3. **User-Interface Entwurf**: Auf Basis der Ergebnisse der vorherigen Analyse wird in dieser Phase das System gestaltet. Die identifizierten und verfeinerten Anforderungen dienen als Vorlage, um passende und konkrete Benutzeroberflächen umzusetzen. Die beteiligten Benutzer werden damit erstmals mit konkreten, sichtbaren Entwürfen konfrontiert. Durch die Berücksichtigung der Anforderungen soll letztlich eine aufgabenangemessene Benutzerschnittstelle entworfen werden, die auf die vorher definierten Usability-Ziele abgestimmt ist. Dieser Schritt kann jedoch nur dann einen erfolgreichen Endentwurf liefern, wenn die vorherigen Entwürfe unter dauerhafter Prüfung der Benutzer standen und die dabei gefundenen Mängel in neuen Entwurfsprozessen eingearbeitet wurden [Off13f].

4. **Evaluationen und Tests**: Zum einen werden hier die elektronischen Prototypen aus dem User-Interface-Entwurf in das Zielsystem überführt und zum anderen werden weiterhin Evaluationen und Tests durchgeführt. Dies soll sicherstellen, dass alle getroffenen Entwurfsentscheidungen im realen System umsetzbar sind [Off13g].

5. **Überleitung in die Nutzung**: In dieser Phase findet die Überleitung des entwickelten Systems in die Nutzung statt. Der Auftraggeber wird dabei begleitet und erhält zusätzlich eine Bedienanleitung. Außerdem findet am Ende meist eine Benutzerschulung statt, die durch die gut dokumentierten Anforderungen und Entwicklungen leicht erstellt werden kann. Mit der Überleitung endet gleichzeitig die Unterstützung der eigentlichen Systementwicklung [Off13g].

6. **Nutzung und Pflege**: Die Phase der Nutzung und Pflege stellt eine Besonderheit des Usability Engineerings dar. Obwohl die ergonomische Begleitung abgeschlossen ist, werden in dieser Phase nochmals Evaluationen durchgeführt, um Rückmeldungen der Endbenutzer zu bekommen. Durch die Nutzung des Systems können Benutzer nützliche Hinweise für eine Systemverbesserung oder -erweiterung liefern. Sind einige Punkte vorhanden, bildet diese Phase die Grundlage für den Start in eine neue Entwicklung. Somit ist diese Phase gleichzeitig Endpunkt der alten und Startpunkt einer neuen Entwicklung [Off13g].

Für eine visuelle Darstellung gibt es zusammenfassend also einige sowohl psychologische als auch designtechnische Vorgaben und Einschränkungen. Man sollte sich über die Funktionen und Vorteile von Bildern bewusst sein und dabei die Kapazität des Arbeitsgedächtnisses berücksichtigen. Hinsichtlich der eigentlichen Gestaltung von Benutzeroberflächen und der Mensch-Maschine-Interaktion gilt die Normreihe ISO 9421, die unter anderem die wichtigsten Grundsätze der Dialoggestaltung beinhaltet, als wichtiger Bezugspunkt. Zusätzlich existieren Modelle wie das ABC-Modell und das Schalenmodell der ganzheitlichen Gestaltung, welche die Notwendigkeit der Aufgaben- und Benutzerorientierung untermauern. Für die Entwicklung der visuellen Ansätze in dieser Arbeit wurde sich für das Vorgehen nach dem Referenzmodell der Daimler AG entschieden. Anhand dieses Modells wird im Verlauf dieser Arbeit auf die folgenden Phasen und Prozessschritte eingegangen (vgl. Abbildung 3.7):

- **Anforderungsanalyse**: Geschäfts-/Einsatzziele, Analyse Ist-Stand, Benutzerprofilanalyse, Aufgabenanalyse und Umgebungsbedingungen
- **User-Interface Entwurf**: User-Interface Mock-Ups, Detailentwurf, Elekronische UI-Prototypen
- **Evaluationen und Tests**

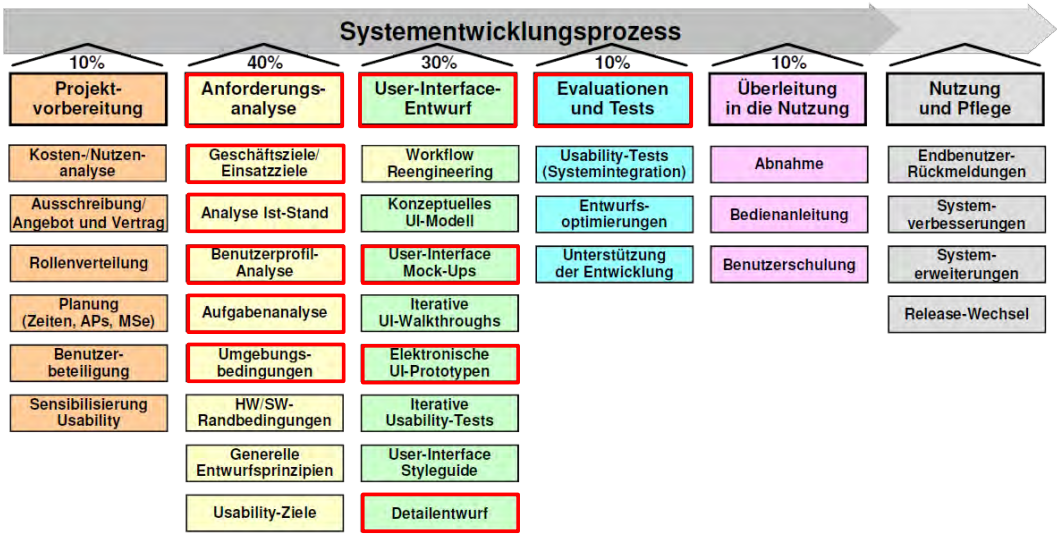


Abbildung 3.7: Alle für diese Arbeit relevanten Phasen und Prozessschritte.

Aus diesen Schritten resultieren zwei visuelle Ansätze, die jedoch auf Grund von Zeit und fehlenden Nutzern nicht in Zusammenarbeit mit den Anwendern entwickelt werden. Zur Überprüfung der technischen Umsetzbarkeit werden beide Ansätze am Ende der Entwicklung mit Hilfe eines Mockup- und Wireframe-Tools umgesetzt. Basierend auf all den in diesem Kapitel erörterten Fakten und dem Referenzmodell für Usability Engineering wird zunächst im folgenden Kapitel die Anforderungsanalyse für Datenabfragen im PROCEED-Framework erläutert.

# 4 Anforderungsanalyse

Wie schon in Abschnitt 3.9 beschrieben, ist die Analysephase die wichtigste Phase des gesamten Referenzmodells des Usability Engineerings. In ihr werden die grundlegenden Anforderungen an das zu entwickelnde System identifiziert. Die Entwicklung eines Systems wird auf Basis von Geschäfts- und Einsatzzielen bestimmt. Dabei werden unter anderem vorhandene Altsysteme analysiert, um Erkenntnisse und Randbedingungen bereits existierender Systeme zu identifizieren und aufzubereiten. Zusätzlich folgt eine ausführliche Analyse der beteiligten Benutzer des zu entwickelnden Systems. Dabei wird geschaut, ob es sich um Experten oder Laien handelt und ob diese das System regelmäßig oder nur sporadisch nutzen. Stehen die Eigenschaften der Benutzer fest, werden deren Aufgaben betrachtet. Jede einzelne Aufgabe wird dabei in Teilschritte aufgegliedert, um detaillierte Kenntnisse zu erhalten. Zusätzlich werden Umgebungsbedingungen untersucht, da unterschiedliche Einsatzorte des Systems (z.B. im Büro, an öffentlichen Orten) weitere Anforderungen an das System mit sich bringen. Auf eine Analyse von Soft- und Hardware, genereller Entwurfsprinzipien und auf eine Festlegung von Usability-Zielen wird in dieser Arbeit jedoch verzichtet. Die Visualisierung von Datenabfragen ist nur ein Teil des gesamten PROCEED-Frameworks und liegt somit im Rahmen einer eigenen umfassenden Analyse, in der auch diese Punkte betrachtet werden. Im Folgenden werden die für diese Arbeit relevanten Prozessschritte der Anforderungsanalyse (vgl. Abbildung 4.1) auf Basis des Usability-Referenzmodells der Daimler AG entwickelt. Insbesondere werden in diesem Abschnitt Anforderungen an das System abgeleitet, die die Basis für die Entwicklung der visuellen Ansätze darstellen.

## 4.1 Geschäfts-/Einsatzziele

Der Sinn und Zweck einer visuellen Darstellung von komplexen E/E-Produktdaten wurde in den vorherigen Kapiteln ausführlich beschrieben. Ziel dieser Darstellungsart ist eine

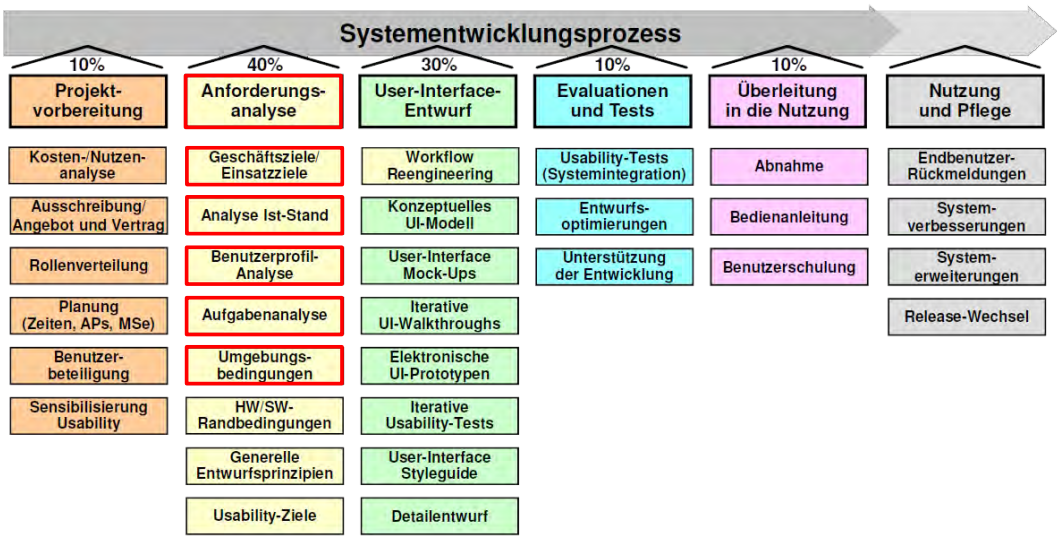


Abbildung 4.1: Für dieses Kapitel relevante Prozessschritte der Anforderungsanalyse des Referenzmodells der Daimler AG: Geschäfts-/Einsatzziele, Analyse-Ist-Stand, Benutzerprofilanalyse, Aufgabenanalyse und Umgebungsbedingungen.

einfache Bereitstellung der Daten, um die Komplexität ihrer Strukturen deutlich zu reduzieren. Zusätzlich soll es möglich sein, durch die angezeigten Produktdaten navigieren zu können.

## 4.2 Analyse Ist-Stand

Diese Arbeit fließt in die Entwicklung des PROCEED-Frameworks mit ein und soll die bisherige Arbeitsweise durch die Transformation von Produktdaten einer Applikation in eine Local Ontology und die anschließende Integration in die Common Integration Ontology (siehe Kapitel 2) deutlich verbessern. Da bisher keine vergleichbaren Systeme vorhanden sind, kann keine Analyse bezüglich existierender Konkurrenzsysteme durchgeführt werden. Welche Probleme die bisherige Arbeit mit E/E-Produktdaten aufweist, wurde bereits ausführlich in Kapitel 2 beschrieben.

## 4.3 Benutzerprofilanalyse

Da sich diese Arbeit ausschließlich auf die Analyse der integrierten Produktdaten bezieht, gibt es auch nur eine zu betrachtende Benutzerrolle. Die sogenannten *Business User* sind Teil des PROCEED-Frameworks und stellen die Datennutzer auf fachlicher Ebene dar (z.B. *Komponentenverantwortlicher*, *Releasemanager* oder *Testingenieur*). Obwohl diese Nutzer Hauptbereitsteller und Nutzer der Produktdaten sind, ist ihr Ausbildungshintergrund nicht immer IT geprägt. Somit haben *Business User* in der Regel keinerlei Kenntnisse über die Wissensrepräsentation (z.B. durch Ontologies) oder die dadurch geschaffene einheitliche Datenstruktur im PROCEED-Framework. Ihr Hauptaugenmerk liegt auf den Produktdaten.

Aus der Klassifizierung der *Business User* als ungeübte und regelmäßige Benutzer kann man für die Gestaltung der Benutzerschnittstelle folgende Schlüsse ziehen: Neben der Berücksichtigung von *Nützlichkeit* und *Komfort*, welches die wichtigsten Kriterien unabhängig vom Benutzer sind, sollte das System unbedingt eine hohe *Verfügbarkeit* besitzen. Regelmäßige Benutzer dürfen nicht durch Systemabstürze oder zu lange Antwortzeiten von ihrer Arbeit abgehalten werden. Zusätzlich sollte eine Navigation durch die Ergebnisse so gestaltet sein, dass sie vom Benutzer intuitiv durchgeführt werden kann. Ein hohes Maß an *Erwartungskonformität* ist gerade für ungeübte Benutzer ein großer Vorteil, da diese sich zwar mit der fachlichen Materie auskennen, jedoch mit einem neu entwickelten System arbeiten müssen. Somit sollte eine Navigation so durchzuführen sein, wie man es als Benutzer erwarten würde. Ebenfalls wichtig in diesem Kontext sind die Kriterien *Erlernbarkeit*, *Fehlerrobustheit* und *Übersichtlichkeit*. Ist eine Benutzerschnittstelle leicht zu erlernen und übersichtlich gestaltet, spielt die fehlende Routine eines Anwenders keine Rolle mehr, da dieser sich durch Beachtung der genannten Kriterien meist genauso gut zu Recht findet wie ein erfahrener und geübter Benutzer.

Für die Gestaltung und Navigation eher irrelevant sind Grundsätze wie *Individualisierbarkeit* und *Steuerbarkeit*. Da die *Business User* ungeübt sind, kommt es ihnen weniger darauf an, sich ihren individuellen Arbeitsbereich zu erstellen, als ihre Aufgabe erledigen zu können.

## 4.4 Kontextuelle Aufgabenanalyse

Neben Reports und Konsistenzüberprüfungen ist die wesentliche Hauptaufgabe der *Business User* die Analyse der Produktdaten. Der Fokus dieser Arbeit liegt somit auf der reinen Informationsbereitstellung. Dabei soll das System bestimmten Anforderungen genügen, die in diesem Abschnitt analysiert werden.

Durch entsprechende Datenabfragen sollen die *Business User* jegliche Information bekommen, die sie vom System haben möchten. Zur Erinnerung: Ergebnisse von Datenabfragen sind Teilausschnitte der Common Integration Ontology, d.h. was die Benutzer zentral dargestellt bekommen sollen, sind die abgefragten Produktdaten aus der CIO (*Anforderung 1*). In Abbildung 4.2 beispielsweise besteht das Ergebnis der Datenabfrage (links) aus den Artefakten (Schema-Concepts oder Individuals) *Fuel Pump*, *Engine*, *Head Lamp*, *Door Driver* und *Door Passenger*.

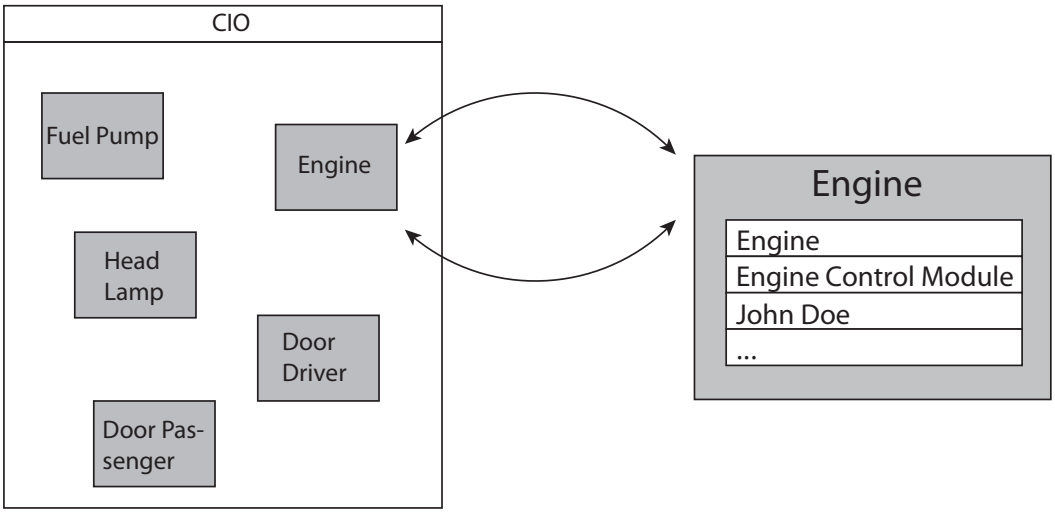


Abbildung 4.2: CIO-Anzeige mit Beispielergebnissen links und einem Beispielelement mit dazugehörigen Attributen rechts.

Da die Artefakte Attribute innerhalb einer Attributliste enthalten, soll es mit dem PROCEED-Framework ebenfalls möglich sein, auf diese Informationen zuzugreifen (*Anforderung 2*). Im Beispiel von Abbildung 4.2 ist dafür exemplarisch das Objekt *Engine* mit dessen Attributliste auf der rechten Seite hervorgehoben. Jedoch gilt es nicht nur zu entwickeln, wie die einzelnen Elemente auszusehen haben, sondern auch, wie diese Elemente als Ge-

samtergebnis dargestellt werden. Somit muss auch eine geeignete Gesamtdarstellung aller Ergebniselemente geschaffen und angezeigt werden (*Anforderung 3*). Dabei wird in dieser Arbeit grundsätzlich zwischen einer geordneten (z.B. ein-/mehrspaltige Liste, Anordnung innerhalb einer geometrischen Form) und einer ungeordneten, also willkürlichen Anordnung innerhalb des Darstellungsraumes unterschieden. Da Software-Anwendungen in eigenen Fenstern gestartet werden und wiederum neue Fenster generieren oder die Verwendung von Tabs ermöglichen, muss in diesem Kontext auch darauf Bezug genommen werden (*Anforderung 4*). Hierbei müssen vor allem die Wichtigkeit der Überschaubarkeit und Vergleichsmöglichkeiten miteinbezogen werden. Fenster kann man beispielsweise mehrfach auf dem Bildschirm anordnen, wodurch der *Business User* eine gute Vergleichsmöglichkeit der einzelnen Fensterinhalte erlangt. Eine Verwendung von einzelnen Tabs vermeidet jedoch das Öffnen und Vorhandensein vieler Fenster auf dem Bildschirm, wodurch die Benutzeroberfläche wesentlich aufgeräumter ist. Aufgrund der Tatsache, dass alle Elemente der CIO aus Local Ontologies integriert wurden, besitzen einige der Artefakte Mappings zu korrespondierenden Elementen aus verschiedenen Local Ontologies (vgl. Kapitel 2). Da in die CIO nicht jedes Element und jedes Attribut eines Elements bei der Integration übernommen wird, beinhalten die LOs meist zusätzliche Informationen (vgl. Abbildung 4.3). Diese

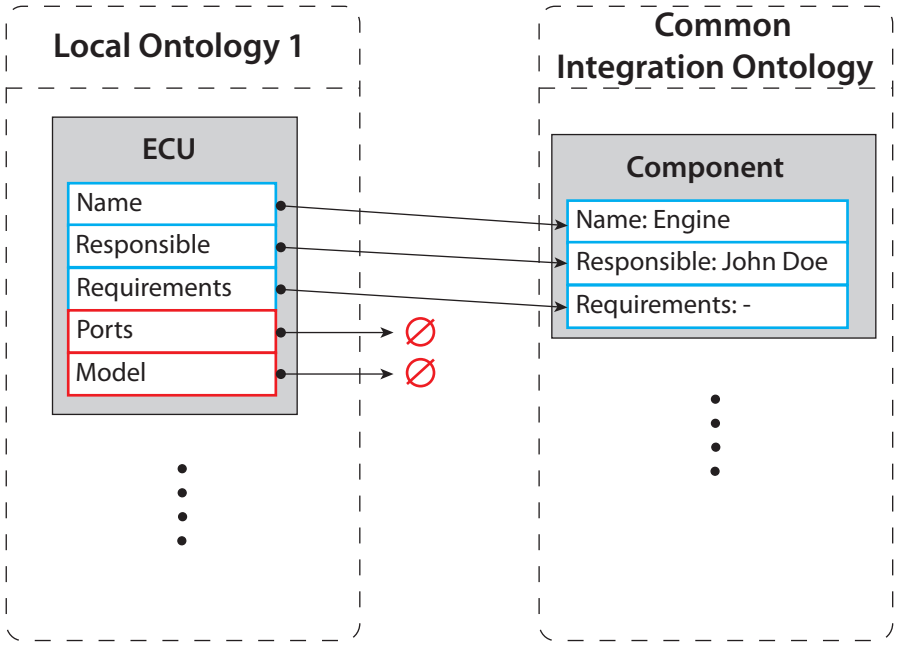


Abbildung 4.3: Attributverlust bei Integration von LO nach CIO.

zusätzlichen Informationen können für die Analyse der Produktdaten von entscheidender Bedeutung sein. Die Local Ontology 1 aus Abbildung 4.3 beinhaltet unter anderem das Schema-Concept *ECU* mitsamt den Attributen *Name*, *Responsible*, *Requirements*, *Ports* und *Model*. Nach der Integration in die CIO besitzt das korrespondierende Schema-Concept *Component* jedoch nur noch die Attribute *Name*, *Responsible* und *Requirements*, während die Attribute *Ports* und *Model* in der CIO unberücksichtigt bleiben. Somit muss es neben der Anzeige der Ergebnisse aus der CIO auch die Möglichkeit geben, die betroffenen Elemente der LOs zu sehen und auf diese zugreifen zu können. Das System muss also sowohl die CIO als auch die korrespondierenden LOs bei der visuellen Darstellung berücksichtigen (*Anforderung 5*). Dabei soll zusätzlich gewährleistet werden, dass *Business User* durch alle angezeigten Elemente navigieren können. Dies bedeutet zusätzlich, dass er in der Lage sein soll, sich durch die vier Datenebenen Product Data Collection, Objects, Variants und Versions zu navigieren. Für die Entwicklung der Darstellung muss also gelten, dass das System eine Möglichkeit bieten muss, dem Benutzer sowohl die Objekte anzuzeigen als auch über einfache Interaktionen Varianten und Versionen darzustellen (*Anforderung 6*). Tabelle 4.1 stellt noch einmal die wesentlichen Anforderungen dar, die das System hinsichtlich der Datenabfragen ermöglichen soll:

| | |
|---|---|
| ***Anforderung 1*** | Darstellung der einzelnen abgefragten Objekte aus der CIO |
| ***Anforderung 2*** | Ansicht der Artefaktattribute |
| ***Anforderung 3*** | Geeignete Gesamtdarstellung aller Ergebniselemente |
| ***Anforderung 4*** | Geeigneter Rahmen für die Gesamtdarstellung |
| ***Anforderung 5*** | Korrespondierende Elemente aus Local Ontologies berücksichtigen |
| ***Anforderung 6*** | Ebenenwechsel zu Varianten und Versionen |

Tabelle 4.1: Anforderungen an das PROCEED-Framework bzgl. Datenabfragen.

## 4.5 Umgebungsbedingungen

Das PROCEED-Framework wird unter üblichen Bürobedingungen auf Desktop-Rechnern ausgeführt. Daher sind keinerlei besonderer Einschränkungen für die Entwicklung gegeben.

# 5 Übertragung der Systemanforderungen auf die Gestaltung

Nachdem im vorherigen Kapitel die Anforderungen für die Visualisierung und Navigation komplexer Produktdaten erörtert wurden, gilt es in diesem Kapitel, diese Anforderungen auf die Gestaltung zu übertragen und dazu passende UI-Mockups zu erstellen (vgl. Abbildung 5.1). Je nach Anforderung lassen sich mehrere Ansätze finden, die jeweils Vor- und Nachteile

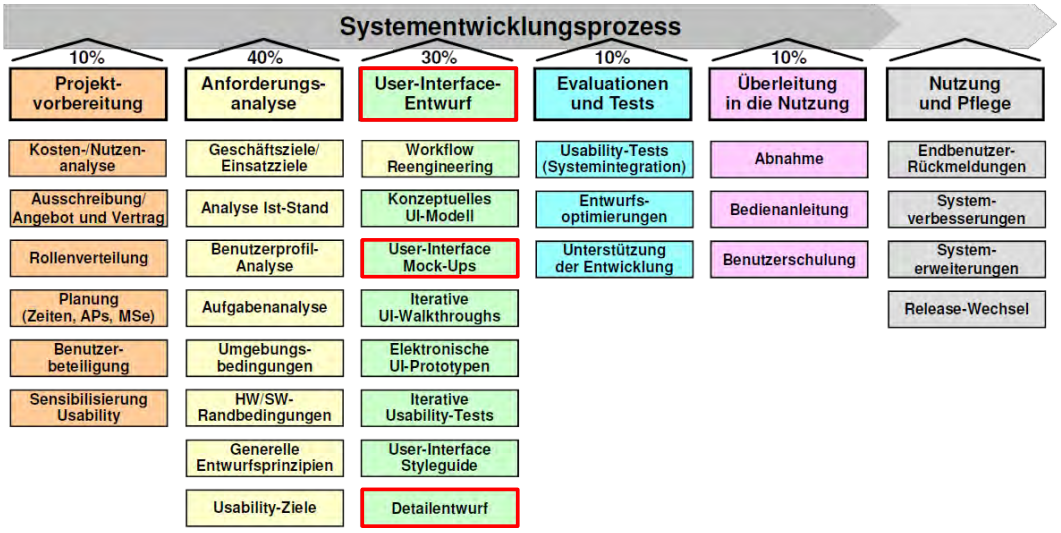


Abbildung 5.1: Für die Kapitel 5, 6 und 7 relevante Prozessschritte der Phase User-Interface-Entwurf des Referenzmodells der Daimler AG: User-Interface Mock-Ups und Detailentwurf.

mit sich bringen. Dabei gilt es für die finalen visuellen Ansätze abzuwägen, welche Seite davon im Einzelfall überwiegt und was dies für Konsequenzen für die Visualisierung und Navigation hat. Darauf aufbauend werden Detailentwürfe erstellt (vgl. Kapitel 6 und 7) und

diese anhand von elektronischen UI-Prototypen auf ihre technische Umsetzbarkeit geprüft (vgl. Kapitel 8).

## 5.1 Anforderung 1: Darstellung der einzelnen abgefragten Objekte aus der ClO

Bei der ersten Anforderung kommt es darauf an, wie man jedes einzelne abgefragte Element einfach, lesbar und übersichtlich darstellen kann. Worauf es dem Benutzer ankommt, ist eine Anzeige aller Ergebniselemente, die er möglichst auf einen Blick betrachten kann (siehe Abschnitt 5.3). Somit muss für die einzelnen Elemente gelten, dass so viele Informationen wie nötig, aber so wenig wie möglich über sie angezeigt werden. Für einen ersten Überblick sollte es demnach ausreichen, dem Benutzer die Namen der einzelnen Artefakte anzuzeigen. Um die schlichten Textfelder visuell ansprechender zu machen, bietet es sich an, die Namen zusammen mit einer einfachen geometrischen Form (z.B. Quadrat, Rechteck, Kreis oder Dreieck) darzustellen (vgl. Abbildung 5.2 (oben)). Wahlweise ist statt einer zweidimensionalen auch eine dreidimensionale Darstellung wie in Abbildung 5.2 (unten) denkbar. Das

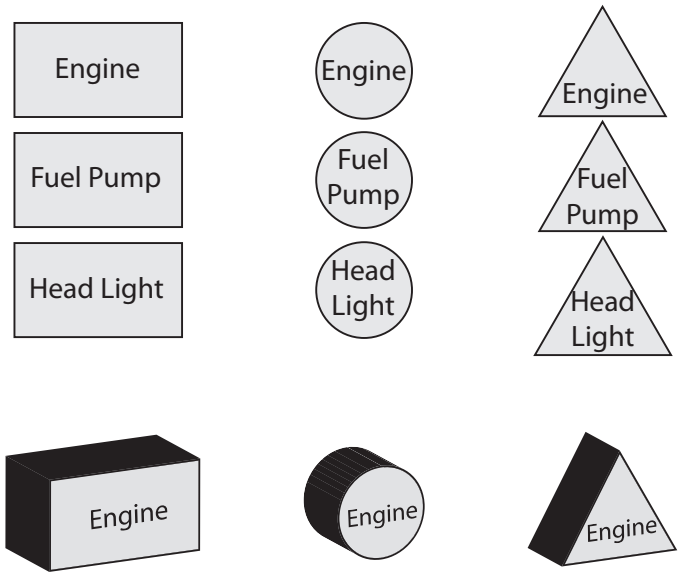


Abbildung 5.2: Zweidimensionale Darstellung der Elemente oben und 3D-Ansicht der Ergebniselemente unten.

Ziel der Darstellung der abgefragten Objekte aus der CIO muss folglich eine einfache, aber klar strukturierte und übersichtliche visuelle Anzeige sein. Unter Umständen ist es sinnvoll, unterschiedlichen Elementen eine unterschiedliche Form zu geben. Eine andere Möglichkeit ist die Verwendung unterschiedlicher Hintergrundfarben der Elemente, falls sich dies nicht mit Lösungen weiterer Anforderungen überschneidet.

## 5.2 Anforderung 2: Ansicht der Artefaktattribute

Jedes Artefakt (Schema Concept, Individual) besitzt auf allen Ebenen eine Menge von Attributen (z.B. *Name*, *Responsible* oder *Description*), die das Artefakt näher beschreiben. Damit der *Business User* auf diese Informationen zugreifen kann, muss das PROCEED-Framework ihm diese Informationen entweder standardmäßig anzeigen oder ihm eine erweiterbare Ansicht anbieten. Der Vorteil einer standardmäßigen Anzeige des Namens sowie der zugehörigen Attribute (vgl. Abbildung 5.3, Bild 1) ist der schnellere Zugriff auf alle Informationen des einzelnen Artefakts. Dies geschieht jedoch auf Kosten der Einfachheit und Übersichtlichkeit, denn je mehr Informationen dargestellt werden müssen, desto voller wird die Anzeige selbst für wenige Ergebniselemente.

Im Gegensatz dazu löst eine erweiterbare Ansicht (vgl. Abbildung 5.3, Bild 2, 3 und 4) die Anforderung wesentlich übersichtlicher, denn in ihr sieht man anfangs nur die Namen der jeweiligen Objekts. Möchte der *Business User* in diesem Kontext nähere Informationen zu den Elementen, so ist eine Interaktion mit dem System nötig. Erst durch eine Aufforderung seitens des Benutzers werden ihm die Attribute eines oder aller Elemente angezeigt. Dabei muss an dieser Stelle zwischen einer *globalen* und einer *elementweisen* Interaktion unterschieden werden. Diese Unterscheidung zwischen globaler und elementweiser Interaktion wird auch in späteren Abschnitten noch von wichtiger Bedeutung sein, da beide Arten Vor- und Nachteile mit sich bringen. Bei einer globalen Interaktion durch Klicken einer angezeigten Schaltfläche (vgl. Abbildung 5.3, Bild 2) ändern sich alle dargestellten Elemente und man sieht von dort an sowohl den Namen als auch eine Attributliste. Diese Ansicht ist dann gleichzusetzen mit der oben genannten standardmäßigen Anzeige der Attribute. Dadurch hat man den Vorteil einer sofort greifbaren, vollkommenen Informationsmenge. Jedoch müssen alle Artefakte automatisch flächenmäßig größer werden, damit in ihnen die Attributliste lesbar angezeigt werden kann. Das Gegenstück dazu bildet die elementweise Interaktion,

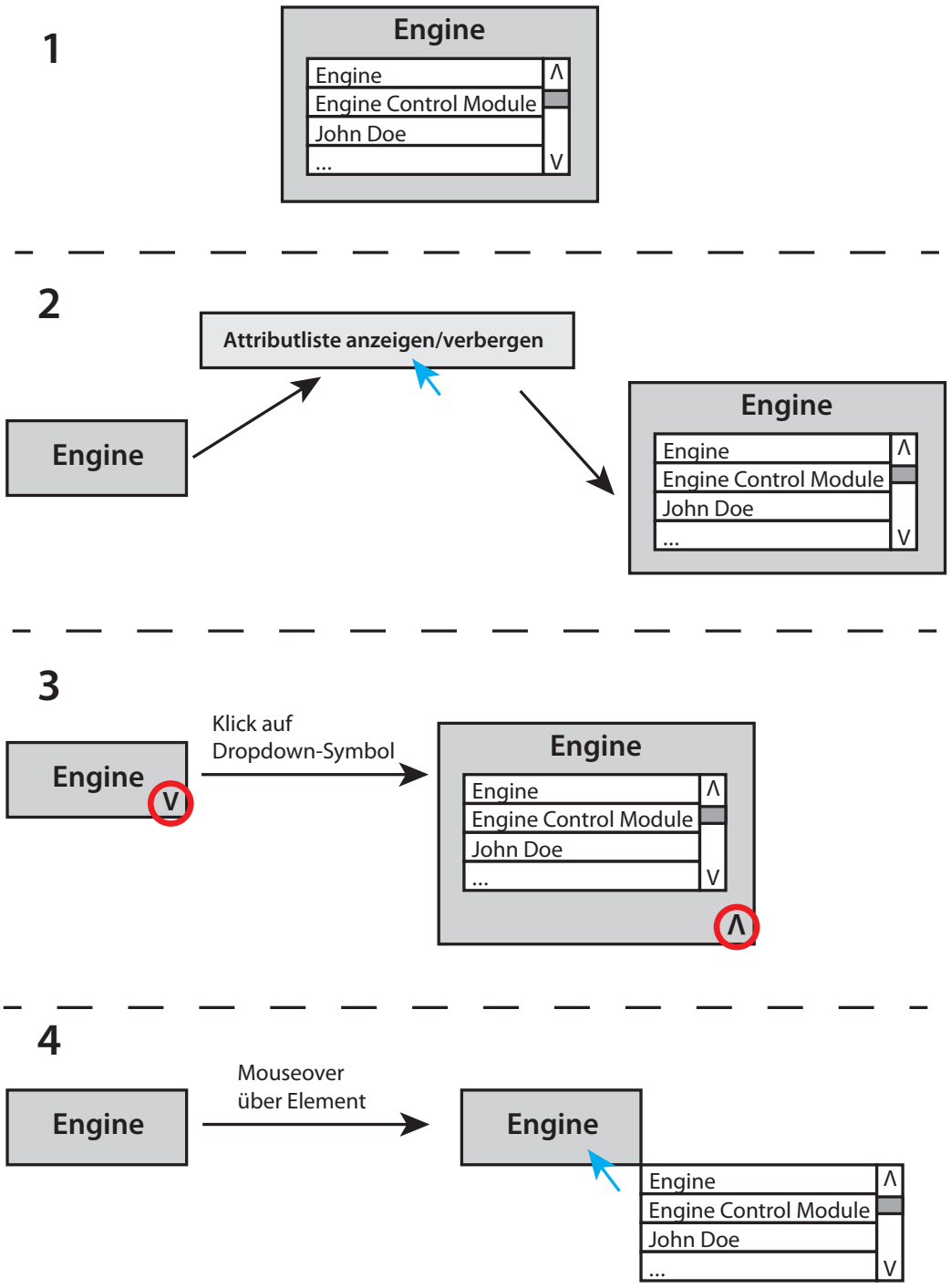


Abbildung 5.3: Verschiedene Möglichkeiten, die Artefaktttribute darzustellen: (1) standardmäßig, (2) durch Klick auf eine globale Schaltfläche, (3) Klick au ein Dropwdown-Symbol am Artefakt oder (4) Mouseover über Artefakt.

bei der der Benutzer lediglich in der Lage ist, einzelne Elemente zu erweitern, um dessen Attribute zu sehen. Dies kann zum Beispiel durch ein Dropdown-Symbol (vgl. Abbildung 5.3, Bild 3) umgesetzt werden. Das Symbol wird an jedem Element platziert und kann durch einen einfachen Mausklick aktiviert werden. Eine andere Möglichkeit ist beispielsweise eine Anzeige der Attribute eines einzelnen Elements mit einem Tooltip bei einem Mouseover oder Mausklick auf eines der Elemente (vgl. Abbildung 5.3, Bild 4). Mit Hilfe der elementweisen Interaktion für die Anzeige der Attribute ermöglicht man dem Benutzer eine aufgeräumte Oberfläche mit den nötigsten Informationen. Gleichzeitig bietet das System aber die Möglichkeit, individuell weitere Informationen ohne viel Aufwand anzuzeigen. Für welche der beiden Interaktionen man sich letztlich entscheidet, hängt von weiteren Entscheidungen ab. Denn je nach Gesamtdarstellung aller Ergebniselemente und dem Darstellungsrahmen, in dem sie angezeigt werden (z.B. Seitenansicht, Tabs oder in eigenen Fenstern), ist eine der beiden Alternativen sinnvoller. Diese noch offenen Punkte werden in den nächsten Abschnitten erörtert.

## 5.3 Anforderung 3: Geeignete Gesamtdarstellung aller Ergebniselemente

Nachdem festgelegt wurde, wie die einzelnen Elemente in etwa auszusehen haben (vgl. Abschnitt 5.1 und 5.2), muss darauf aufbauend eine geeignete Gesamtdarstellung all dieser Elemente entwickelt werden. Wie bereits angemerkt, ist eine einfache, übersichtliche, aber aufgabengerechte visuelle Anzeige das Ziel. Aus Kapitel 2 ist bekannt, dass bei einer umfangreichen Abfrage mehrere hundert Elemente angezeigt werden müssen. Zur Erinnerung: Eine Element-Darstellung besteht aus dessen Namen und eventuell dessen Attributliste. Obwohl heutige Desktop-Bildschirme einen enormen Darstellungsspielraum bezüglich der Oberflächenentwicklung bieten, muss man sich bei der Entwicklung trotzdem ausführlich Gedanken machen. Worauf es bei dieser Anforderung hauptsächlich ankommt, ist eine Darstellung, mit der man eine Vielzahl an Elementen optimal auf dem Bildschirm platziert. Optimal bedeutet in diesem Kontext, dass so viele Elemente wie möglich lesbar dargestellt werden, ohne dass der Benutzer erst lästige Navigationsinteraktionen tätigen muss, um sich einen kompletten Überblick über die Ergebnismenge zu verschaffen. Im Folgenden

werden verschiedene Ansätze vorgestellt, die diese Anforderung auf unterschiedliche Weise erfüllen.

### 5.3.1 Geordnete Struktur

Als erster Ansatz einer geeigneten Gesamtdarstellung aller Ergebniselemente kommt eine geordnete Struktur in Frage. Mit dem Begriff geordnet ist in diesem Kontext sowohl die reine Anordnung der Elemente als auch eine gewisse Reihenfolge gemeint.

- **Idee 1: Einspaltige Liste**: Die erste und einfachste Idee für eine Darstellung einer Menge von Dingen ist eine einspaltige Liste wie die einer Einkaufs- oder To-Do-Liste. Für eine überschaubare Menge an Produkten oder Aufgaben ist diese Lösung völlig ausreichend. Auf einen Blick sieht man alle Listenelemente und kann diese analysieren. Sobald man jedoch eine Vielzahl von Produktdaten in der Liste hat, verliert man schnell den Überblick. Was also für wenige Elemente klappt, kann schon für mehrere Elemente ein Problem werden. Da es durchaus vorkommen kann, dass viele Elemente als Ergebnis geliefert werden, kommt eine einspaltige Liste für die visuelle Ergebnisanzeige im PROCEED-Framework als allgemeine Lösung des Problems nicht in Frage. Selbst eine seitliche Ausrichtung der Liste aufgrund des Seitenverhältnisses der meisten Bildschirme wird der Anforderung für mehrere Elemente kaum gerecht werden. All dies ist nur auf Kosten der Lesbarkeit der Elemente umsetzbar, welche jedoch durch *Anforderung 1* eingehalten werden muss.

- **Idee 2: Zweispaltige Liste**: Um dem Problem der ersten Idee entgegenzuwirken, könnte man statt einer einzigen Spalte eine zweispaltige Darstellung verwenden. Dies ergäbe zwar eine nur noch halb so lange Liste, erfordert bei der Vielzahl an möglichen Ergebnissen jedoch ebenfalls einen Mehraufwand, um alle Elemente zu überblicken. Letztlich ist die Idee einer Auflistung jedoch praktisch und erweiterbar, weshalb eine weitere Idee in Betracht gezogen wird: eine mehrspaltige Liste.

- **Idee 3: Mehrspaltige Liste**: Die Herangehensweise mit einer mehrspaltigen Liste wird das Darstellungsproblem selbst für eine Vielzahl an Elementen im passenden Zusammenspiel mit dem äußeren Rahmen (siehe Abschnitt 5.4) verhältnismäßig zufriedenstellend lösen. Zufriedenstellend bedeutet in diesem Zusammenhang ein

angemessenes Verhältnis zwischen Überschaubarkeit der Elemente und Navigationsaufwand. Außerdem impliziert der Begriff mehrspaltig automatisch auch eine Darstellung einer zweispaltigen Liste, womit auch auf kleine Ergebnismengen Rücksicht genommen wird. Im Falle nur sehr weniger Elemente kann jedoch durchaus auch auf eine einspaltige Lösung zurückgegriffen werden. Diese dritte Idee vereinigt somit die Vorteile der ersten beiden Ideen und löst gleichzeitig deren Probleme bezüglich mehrerer Elemente.

Das eine solche Lösung sinnvoll ist, beweisen in der Praxis einige Anwendungen, die ebenfalls mit einer Vielzahl von darzustellenden Elementen zu kämpfen haben. Neben Datei-Explorern, die jegliche Arten von Dateien und Ordnern meist in einer mehrspaltigen Liste anzeigen, dient Microsofts Betriebssystem Windows 8 mit dessen Kachel-Oberfläche als gutes Beispiel. Im Microsoft Store befinden sich derzeit ca. 150.000 verschiedene Apps [Win14]. Ungeachtet der Möglichkeit, die Apps in verschiedenen Größen darzustellen, werden alle heruntergeladenen Apps als einzelne Kacheln gemeinsam auf der Startoberfläche dargestellt. Diese Fläche lässt sich ein Stück weit nach unten, aber vor allem seitlich erweitern, sodass man mehr Platz für die Apps hat. Um durch die Ansicht zu navigieren, benötigt man in diesem Fall je nach Berücksichtigung weiterer Anforderungen beispielsweise lediglich eine Scrollbar an den Rändern der Oberfläche oder eine Schaltfläche, um eine Anzeigen-Seite weiter zu blättern (vgl. Abschnitt 5.4). Zu einer solchen Darstellung gehört automatisch auch eine bestimmte Reihenfolge, in der die einzelnen Elemente angeordnet werden müssen. Dabei unterscheidet man zwischen einer chronologischen, alphabetischen und einer durch Zählen definierte Reihenfolge [Wikc]. Im Falle des PROCEED-Frameworks und der Produktdaten kommen beispielsweise folgende Reihenfolgen in Frage:

- **Alphabetische Reihenfolge**:
  - Die Menge der Elemente wird anhand der Objektnamen alphabetisch geordnet.
  - Einzelne Attributwerte können ebenfalls für eine alphabetische Reihenfolge hergezogen werden. Falls diese jedoch nicht gleich erkenntlich sind (vgl. *Anforderung 2*), könnte dies den Benutzer eventuell verwirren.
- **Chronologische Reihenfolge**: Elemente, die erst kürzlich hinzugefügt wurden, könnten vom System als erste Elemente dargestellt werden. Dadurch entsteht eine chronologische Reihenfolge der Elemente, in der der *Business User* direkt erkennen kann,

welche Elemente neu und welche schon älter sind. Diese Art der Reihenfolge erfordert jedoch über alle Individuals einheitlichen Attributwert wie „Datum“ oder „Erstelldatum“ oder Ähnliches. Dies verpflichtend zu verlangen, nur um eine Reihenfolge definieren zu können, ist eher keine gute Lösung. Da Individuals aus unterschiedlichen Local Ontologies stammen, ist es eher unwahrscheinlich, ein solches Attribut zu finden, welches in allen Systemen auch im gleichen Format gepflegt wurde.

- **Reihenfolge durch Zählen**: Da eine Reihenfolge, die auf Zahlen basiert, gleichzeitig auch immer diese Zahlen vorangestellt haben sollte, macht diese Art der Reihenfolge für die Auflistung von Ergebniselementen einer Datenabfrage wenig Sinn und würde nur unnötig Darstellungsraum verschwenden.

- **Beliebige Reihenfolge**: Statt die Elemente an irgendeine Reihenfolge zu binden, ist es auch denkbar, die Elemente in einer beliebigen Reihenfolge anzuordnen. Aufgrund der fehlenden Orientierungsmöglichkeit des Benutzers bei einer Anordnung mit keinerlei Reihenfolge, könnte die Analyse der Datenabfrage schwerer fallen als bei der Wahl einer konkreten Reihenfolge. Wenn man als Benutzer keinerlei Kriterium feststellen kann, nach welchem sich die dargestellte Menge richtet, brauchen Suchvorgänge nach einzelnen Elementen auf dem Bildschirm womöglich entsprechend länger. Da das System jedoch nach Usability-Richtlinien entworfen wird, ist eine solche Lösung für eine gute Benutzbarkeit nicht geeignet. Dies müsste jedoch im entsprechenden Fall getestet und evaluiert werden.

### 5.3.2 Geometrische Darstellungsform

Ähnlich zu der in Abschnitt 5.3.1 rechteckigen Darstellung ist es denkbar, andere geometrische Formen zu verwenden. Zum Beispiel kommen dabei Dreiecke, Kreise und Sechsecke in Betracht. Zu Beginn der Überlegung macht es Sinn, den Flächeninhalt der genannten Formen zu betrachten. Denn Ziel des Ganzen ist und bleibt eine übersichtliche Darstellungsart, durch die so viele Elemente wie möglich auf der Bildschirmoberfläche angezeigt werden sollen. Somit werden diejenigen Formen gesucht, die einen großen Flächeninhalt aufweisen. Das Diagramm in Abbildung 5.4 vergleicht den Flächeninhalt eines gleichseitigen Dreiecks mit einem Kreis und einem Rechteck als Richtwert. Die Längen der beiden Seiten des Rechtecks wurden dabei im Verhältnis 3:2 gewählt. Eine Analyse des Diagramms ergibt,

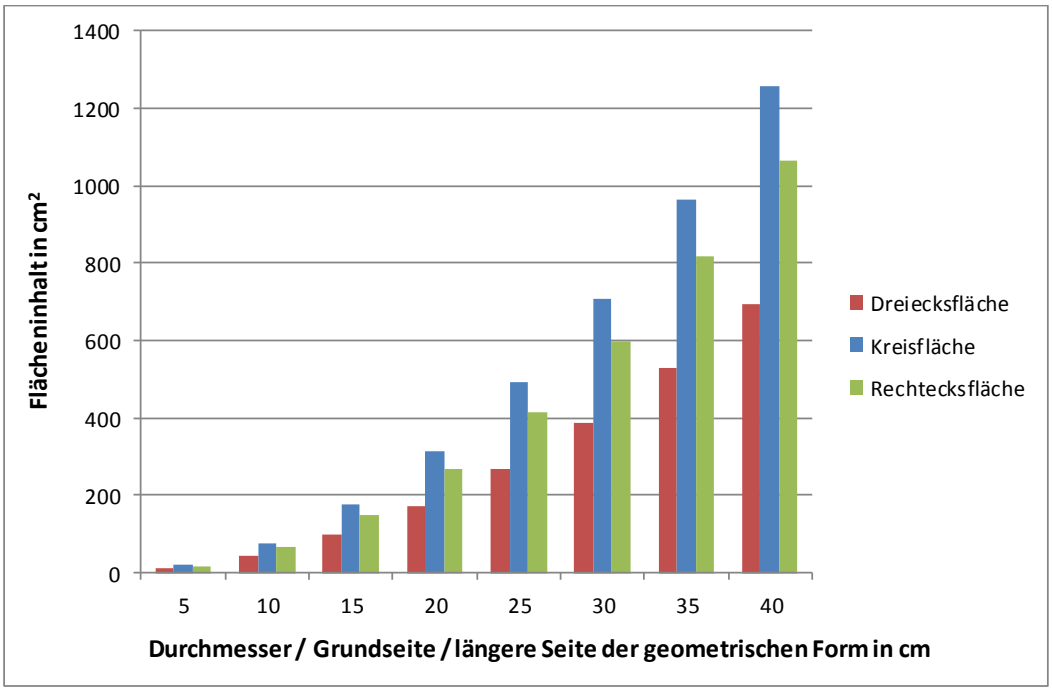


Abbildung 5.4: Darstellung des Flächeninhalts je nach Durchmesser/Länge der Grundseite der geometrischen Form.

dass die Wahl einer Dreiecksfläche nur wenig sinnvoll ist, um in ihr so viel Informationen wie möglich anzuzeigen. Außerdem resultiert aus dem Diagramm, dass es letztlich kaum einen Unterschied macht, ob man sich für eine kreisförmige oder eine rechteckige Fläche entscheidet. Somit kann als Endergebnis dieser Analyse geschlossen werden, dass sowohl Rechtecke als auch Kreise als einzelne Darstellung und auch als Gesamtdarstellung der Ergebniselemente in Frage kommen.

### 5.3.3 Ungeordnete Darstellung

Als Alternative zu den beiden bisherigen Ansätzen wird jetzt eine Idee angesprochen, die sich an keine Ordnung hält. Was damit gemeint ist, spiegelt Abbildung 5.5 wider. Die einzelnen Ergebniselemente werden dabei irgendwie auf der verfügbaren Fläche dargestellt. Durch die beliebige Anordnung kann in diesem Fall auf keine Reihenfolge zur Orientierung zurückgegriffen werden. Nachteilig zu betrachten ist hierbei auch der unnötig verlorene

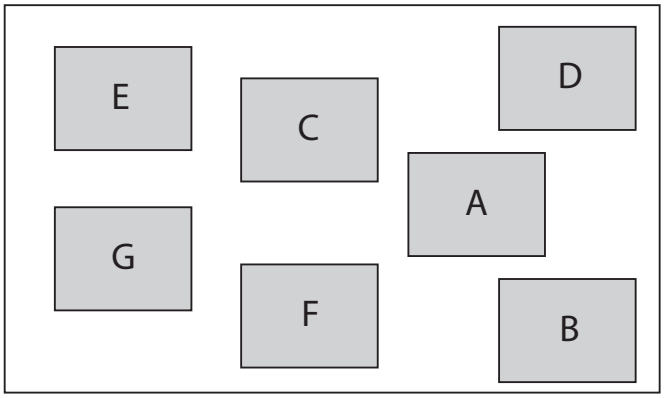


Abbildung 5.5: Ungeordnete Darstellung der einzelnen Ergebniselemente auf der verfügbaren Oberfläche.

Platz, der bei dieser Anordnung entsteht. Andererseits bietet er bei wenigen Elementen automatisch Fläche zur Vergrößerung der Elemente für die Anzeige der Attributliste. Dadurch könnte eine Überdeckung oder Verschiebung geschnittener Elemente verhindert werden, was bei den oben genannten Ansätzen in Kauf genommen werden muss. Bei mehreren Elementen könnte man den Platz zwischen den Elementen sowie deren Größe verringern, sodass eine weitere Anzahl an Elementen ohne Benutzerinteraktionen auf einer einzigen Oberfläche sichtbar ist. Dieser Vorgang kann soweit ausgereizt werden, bis die Elemente gerade noch lesbar sind oder die Darstellung zu einer geometrischen Darstellungsform zusammen geschrumpft ist.

## 5.4 Anforderung 4: Geeigneter Rahmen für die Gesamtdarstellung

Ein weiterer wichtiger Punkt ist der „Rahmen“ für die Gesamtdarstellung. Damit ist eine bestimmte Oberfläche gemeint, in der sich das Gesamtgebilde der Ergebniselemente befinden soll.

### 5.4.1 Verwendung von Fenstern

Üblicherweise werden für solche Zwecke Fenster verwendet, wie z.B. der Datei-Explorer unter Windows. Dabei können die Fenster über den kompletten Bildschirmbereich ragen oder in einer kleineren Form auf dem Bildschirm zu sehen sein. Außerdem können mehrere Fenster auf einem Bildschirm angezeigt werden – beispielsweise nebeneinander, übereinander oder überlappend – wodurch man gleichzeitig den Blick auf mehrere Inhalte richten kann. Dies ist ein entscheidender Punkt bei der Darstellung vor allem zusammengehöriger Elemente. Besonders wenn mehrere Ansichten angeboten werden sollen (vgl. *Anforderung 5* und *Anforderung 6*), ist eine Darstellung von Vorteil, dessen Größe und Position individuell verändert und zusammen mit anderen Anzeigen am Bildschirm platziert werden kann. Der Nachteil einzelner Fenster liegt jedoch auf der Hand: Je mehr Fenster man öffnet, desto voller und unübersichtlicher wird es auf dem Bildschirm. In Bezug auf die Übersichtlichkeit ist dieser Nachteil definitiv bei der Entwicklung miteinzuberechnen. Andererseits hängt dieser Fakt auch sehr vom jeweiligen Benutzer ab. Schließt er nicht mehr benötigte Fenster gleich wieder, erspart er sich das Problem.

### 5.4.2 Verwendung von Tabs

Eine andere Möglichkeit, mehrere Ansichten in einem einheitlichen Rahmen darzustellen, sind Tabs wie sie beispielsweise von den üblichen Webbrowsern verwendet werden. Öffnet man einen Webbrowser, erscheint ein einzelner Tab am oberen Rand, welcher es ermöglicht, auf dieser Anzeige im Internet zu surfen. Möchte man seine aktuelle Webseite angezeigt lassen, aber zum Beispiel eine dazu passende weitere Webseite aufrufen, so kann man einen weiteren Tab daneben öffnen. Somit hat man sowohl die Möglichkeit, die Infos der ersten Seite nicht zu verlieren, als auch die Möglichkeit, zwischen beiden Ansichten hin und her zu navigieren. Sobald mehrere Tabs ins Spiel kommen, wird zu ihrer Darstellung die volle Fensterbreite des Browsers genutzt. Überschreitet die Anzahl der Tabs die Breite, so wird beispielsweise in Google Chrome die Breite der einzelnen Tabs verringert. Der Vorteil dieser Art der Darstellung ist eindeutig die flexible Anzeigennavigation durch mehrere dargestellte Inhalte in einem einzigen geöffneten Fenster. Man hat parallel Zugriff auf mehrere Webseiten, die nur durch einen einfachen Klick auf den entsprechenden Tab voneinander getrennt sind. Außerdem besteht die Möglichkeit, einen einzelnen Tab aus der „Reihe“

zu lösen, wodurch ein eigenes neues Webbrowser-Fenster mit diesem Tab geöffnet wird. Dadurch eröffnet sich zusätzlich der Vorteil der Fensterdarstellung, nämlich das Anordnen mehrerer Fenster auf dem Bildschirm. Damit hat der Benutzer die Möglichkeit, auf einen Blick mehrere Ansichten miteinander zu vergleichen. Für die Aufgabe der Datenanalyse ist dies ein wichtiges Argument. Je nach Anzahl der einzelnen Fenster ist hierbei jedoch wieder das Problem der Übersichtlichkeit zu beachten. Der Nachteil einer Tab-Ansicht ist die Beschränkung der Ansicht auf einen Tab, insofern man sie nicht in einzelne Fenster auslagert. Außerdem wird je nach Anzahl der verschiedenen benötigten Ansichten (vgl. *Anforderung 5* und *Anforderung 6*) die Reihe der Tabs entsprechend lang. Bei vielen Tabs muss durch die Liste gesucht werden, um eine der schon geöffneten Ansichten zu finden. Entscheidet man sich für die Variante der Tabs, so müssen unter anderem diese Punkte dabei berücksichtigt werden.

Zusammenfassend für die beiden Abschnitte 5.4.1 und 5.4.2 ist zu sagen, dass typische Anwendungen auf diese beiden Konzepte (Fenster und Tabs) üblicherweise zurückgreifen. Um den Mitarbeitern keinen unnötigen Aufwand zu generieren, sollten sich Ansätze für das PROCEED-Framework zumindest hinsichtlich der Datenabfragen auf eins der beiden Konzepte konzentrieren. Dabei spricht auch wenig gegen dieses Vorhaben, denn die Vorteile passen gut zu den in Kapitel 4 analysierten Anforderungen und der Nachteil der Unübersichtlichkeit liegt allein am Benutzerverhalten. Nur wenn der *Business User* viele Fenster oder Tabs öffnet, hat dies negativen Einfluss auf die Übersichtlichkeit. Hält er die Anzahl in einem überschaubaren Rahmen, so kann er die genannten Vorteile ohne wirkliche Nachteile nutzen und sich ganz auf seine eigentliche Aufgabe konzentrieren. Außerdem sind im Hinblick auf die beiden genannten Konzepte Erweiterungen (z.B. engere Kombination beider Konzepte oder neue Features) denkbar, die eine individuellere und möglicherweise noch passendere Anwendung hervorbringen.

### 5.4.3 Verwendung von Seiten

Innerhalb der Fenster und Tabs wurde bisher von einer Oberflächenbehandlung ausgegangen, die bei der Darstellung von zu vielen Elementen ein Scrolling vorsieht. Diese Methode ist eine sehr platzsparende und einfache Art und Weise, mit vielen Artefakten umzugehen (vgl. Abschnitt 5.3.1). Alternativ dazu ist auch eine bestimmte Anzahl von Seiten denkbar.

Eine Seite ist dabei wie eine Buchseite (z.B. E-Book) zu verstehen. Sie entspricht beispielsweise der Größe der Oberfläche im Fenster/Tab und enthält so viele Elemente, wie dort hineinpassen. Gibt es mehr Elemente, so wird eine zweite Seite angefügt und der Benutzer kann über eine kleine Schaltfläche durch die Seiten blättern (vgl. Abbildung 5.6).

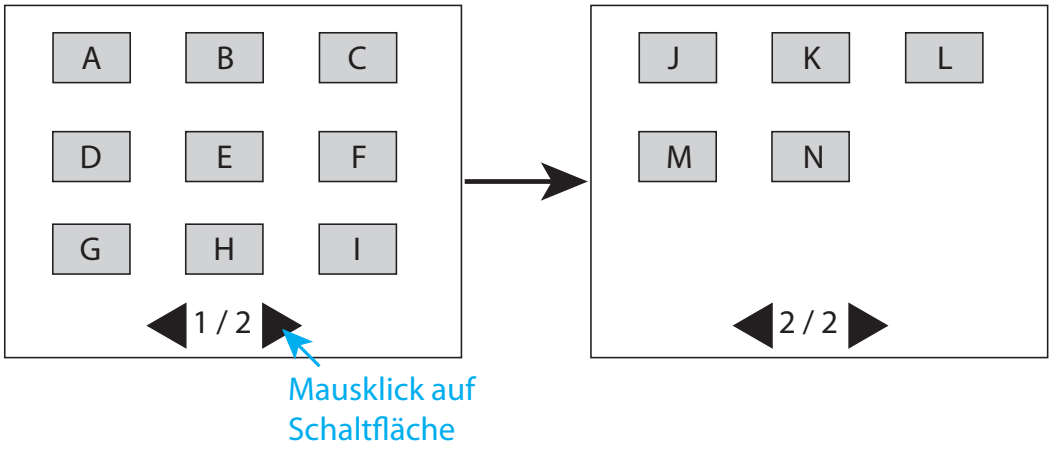


Abbildung 5.6: Seitenansicht der Ergebniselemente. Sind mehrere Seiten verfügbar, so kann der Benutzer über eine Schaltfläche zu den nächsten Seiten navigieren.

## 5.5 Anforderung 5: Korrespondierende Elemente aus Local Ontologies berücksichtigen

Nachdem in den vorherigen Abschnitten die grundsätzlichen Rahmenbedingungen für die Darstellung sowohl der einzelnen Elemente als auch ihrer Gesamtheit in einem passenden Rahmen festgelegt wurden, folgen mit diesem und dem nächsten noch zwei letzte wichtige Abschnitte. Der erste konzentriert sich auf die Berücksichtigung korrespondierender Artefakte (Schema Concepts und Individuals) aus Local Ontologies. Zur Erinnerung: Die Artefakte der Local Ontologies besitzen bestimmte Mappings zu korrespondierenden Elementen in der CIO (vgl. Kapitel 2). Diese Elemente geben Aufschluss über weitere Informationen oder verbundenen Elemente, die in der CIO nicht dargestellt sind. Für die Gestaltungsentwicklung ist nun die Frage, in welcher Art und Weise man die Darstellung dieser Elemente ins System eingliedert und wie man darauf zugreifen kann. Relevante Fragen sind hierbei:

- Wo werden die Elemente dargestellt?
- Welche Informationen werden dargestellt?
- Wie werden Elemente aus Local Ontologies sowie der Common Integration Ontology zusammen dargestellt?

### 5.5.1 Darstellungsort

Die erste wichtige Entscheidung bestimmt, an welchem Ort die korrespondierenden Elemente der Local Ontologies platziert werden. In Frage kommen dabei die Hauptansicht, also die Anzeige der Ergebniselemente oder die Anzeige der korrespondierenden Elemente in einem eigenen Fenster oder in einem eigenen Tab. Entscheidet man sich für die gemeinsame Darstellung in einer Anzeige, so muss man den dadurch erhöhten Platzmangel berücksichtigen (vgl. Abbildung 5.10). Da es sich bei den Ergebnissen schon um eine Vielzahl von Elementen handeln kann, wird die zusätzliche Anzeige der dazugehörigen Elemente der Local Ontologies ein großes Problem und sollte vermieden werden. Eine Auslagerung in ein eigenes Fenster oder eigenes Tab ist dabei die wesentlich geeignetere Lösung (vgl. Abbildung 5.7 und 5.8).

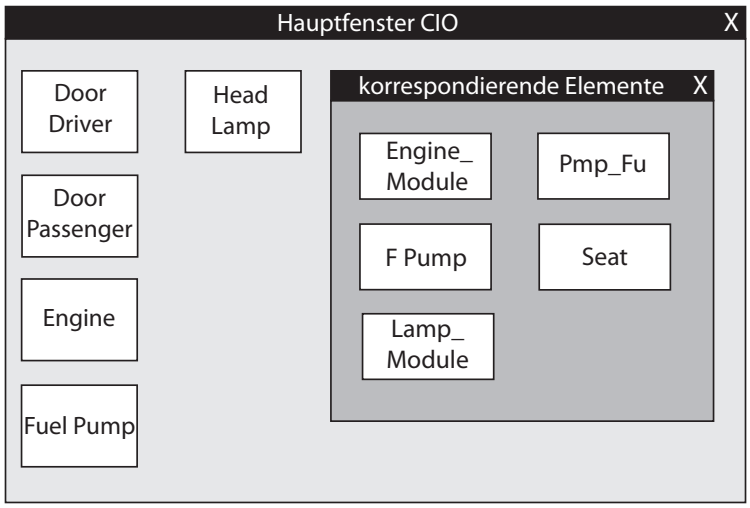


Abbildung 5.7: Auslagerung der korrespondierenden Elemente in ein extra Fenster.

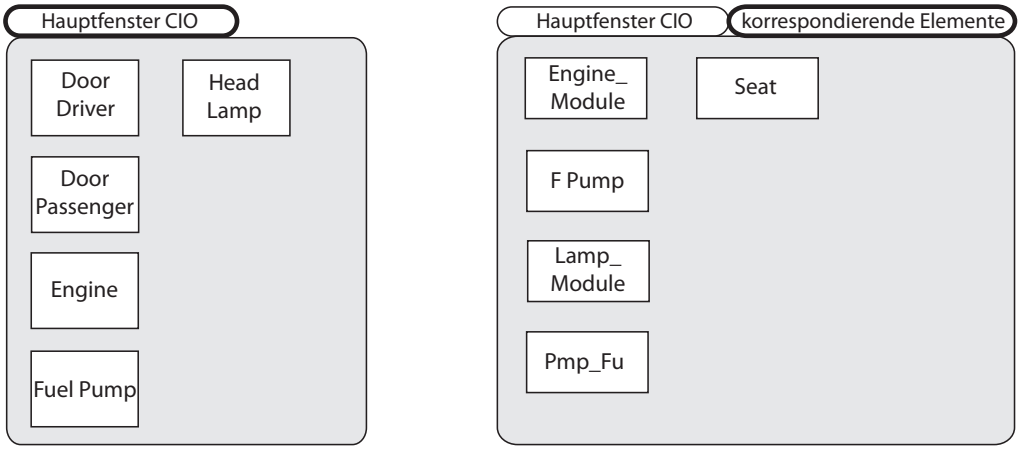


Abbildung 5.8: Auslagerung der korrespondierenden Elemente in ein neues Tab.

Alternativ zu einer Darstellung im CIO-Bereich oder in weiteren Fenstern beziehungsweise Tabs kann beispielsweise auch an eine Anzeige der entsprechenden Elemente in einem Tooltip per Mouseover-Effekt realisiert werden (vgl. Abbildung 5.9).

Während die schon genannten Ansätze eine globale Anzeige aller Elemente repräsentieren, ist diese Tooltip-Lösung eine Variante für eine elementweise Darstellung der korrespondierenden Elemente eines einzelnen Ergebniselements. Je nach Anzahl der korrespondierenden Elemente und Bildschirmgröße muss hierbei eine passende Größe des Tooltips gewählt werden. Wie genau die Elemente darin dargestellt sind, ist ebenfalls zu bedenken. Um das System konsistent zu halten, sollte man sich hierbei an eine Darstellung der Elemente halten, für die man sich in *Anforderung 1* entschieden hat.

### 5.5.2 Darzustellende Informationen

Nachdem feststeht, welche Darstellungsorte für die korrespondierenden Elemente in Frage kommen, muss geklärt werden, welche Informationen dieser Elemente relevant sind. Grundsätzlich gilt, dass der Elementname unbedingt angezeigt werden muss. Der Hauptfokus dieses Punktes liegt viel eher auf der Information, aus welcher Local Ontology das jeweilige Element stammt, also ob der Benutzer diese Information direkt zusammen mit dem korrespondierenden Artefakt angezeigt bekommen muss oder erst nach einer Interaktion mit dem System.

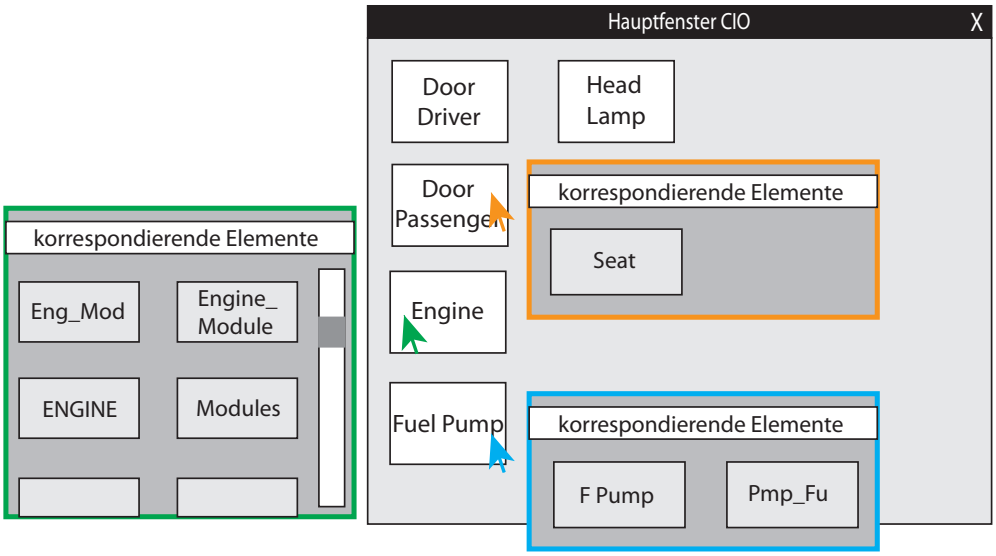


Abbildung 5.9: Verwendung von Tooltips zur Anzeige von korrespondierenden Elementen.

**Fall 1**: Die Abbildung 5.10 fasst die Darstellungsmöglichkeiten zusammen, insofern man sich für eine Anzeige zusammen mit der Ergebnismenge entscheidet.

Links oben in Abbildung 5.10 zeigt das PROCEED-Framework lediglich die betroffenen LOs (*Local Ontology 1-4*), wodurch der Benutzer zwar einen Überblick über die jeweiligen LOs bekommt, aber ohne Interaktion nicht weiß, welche Elemente aus der CIO genau betroffen sind. Da er mit dieser dort fehlenden Information aber wesentlich mehr anfangen kann, bleibt die Wahl zwischen einer reinen Darstellung der korrespondierenden Elemente (Abbildung 5.10, links unten) oder einer gemeinsamen Anzeige (Abbildung 5.10, rechts oben). Da diese sehr viel Platz in Anspruch nimmt, muss diese Darstellungsart mit Vorsicht betrachtet werden. Eine Alternative der drei Möglichkeiten ist die Verwendung einer kleinen Information beim Klick auf eines der Ergebniselemente (Abbildung 5.10, rechts unten). Beim Klick auf das korrespondierende Element *F_Pump* beispielsweise bekommt der Benutzer die kleine textuelle Mitteilung, dass dieses Element aus der LO2 stammt.

**Fall 2**: Bei der Wahl eines Fensters als Darstellungsort der korrespondierenden Elemente (samt LOs) kommen wiederum andere Möglichkeiten in Frage. Die erste Idee ist es, für jede betroffene LO ein eigenes Fenster darzustellen, in dem die dazugehörigen korrespondierenden Elemente der Ergebnis-Artefakte angezeigt werden (vgl. Abbildung 5.11). Der

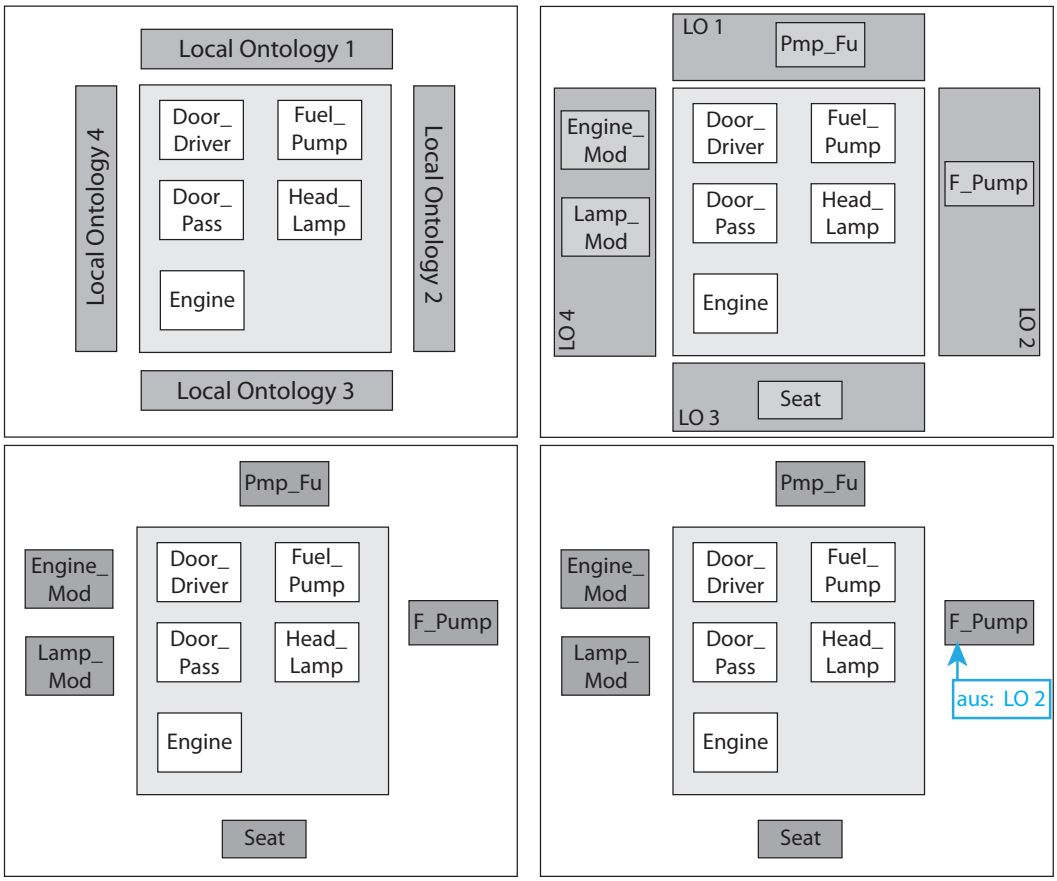


Abbildung 5.10: Möglichkeiten der Darstellung der korrespondierenden Elemente mit oder ohne LO.

Fenstername entspricht dabei jeweils dem Namen der Local Ontology. Die zweite Idee ist ein neues Fenster, in dem für jede Local Ontology ein Tab in diesem Fenster geöffnet ist, welches wiederum die korrespondierenden Elemente enthält (vgl. Abbildung 5.12). Da sich diese beiden Ideen bei ungünstigen Abfragen, in denen eine Vielzahl von LOs betroffen sind, als äußerst unangenehm für den Benutzer herausstellen, ist vor allem das Fensterbeispiel eher als Negativbeispiel zu betrachten. Bessere Lösungen sollten sich auf ein einziges neues Fenster konzentrieren, um den Bildschirm des Benutzers vor allem bei größeren Abfragen nicht zu überfüllen. Dabei kann beispielsweise eine einfache, interne Untergliederung in diesem Fenster als ausreichende Informationsdarstellung genügen. Jede betroffene LO wird zum Beispiel als eine Art Darstellungsfläche angezeigt, in der die dazugehörigen korrespondierenden Elemente positioniert sind (vgl. Abbildung 5.13).

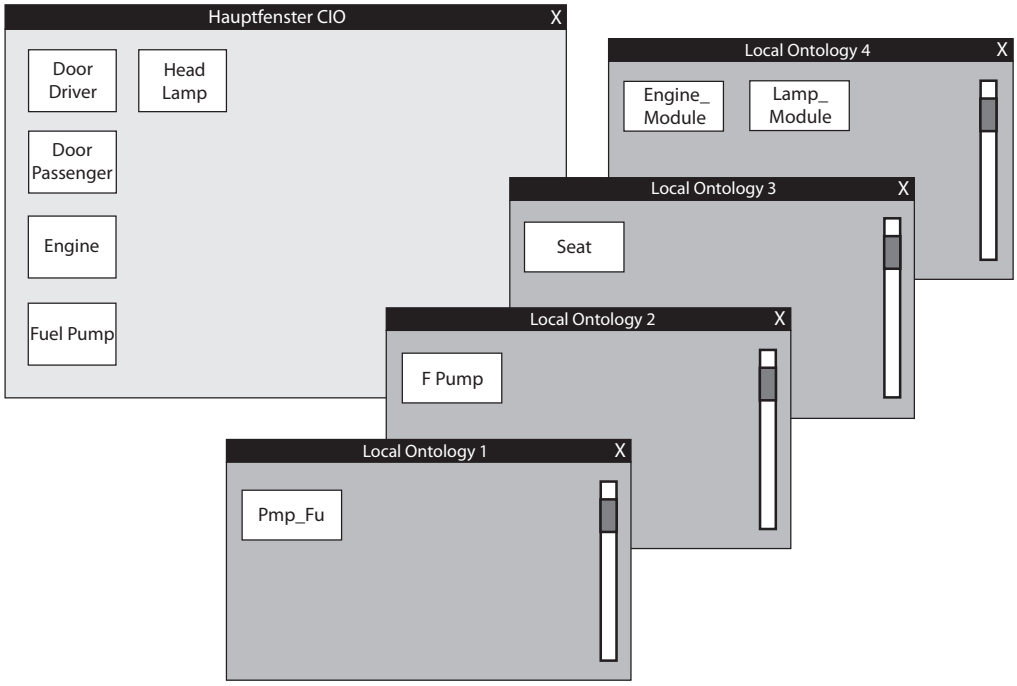


Abbildung 5.11: Für jede LO wird ein eigenes Fenster verwendet, um die dazugehörigen korrespondierenden Elemente anzuzeigen.

**Fall 3**: Entscheidet man sich für einen extra Tab als Darstellungsort, so kommen wie bei dem Fenster-Ansatz ebenfalls keine einzelnen Tabs für jede einzelne LO in Frage. Auch hier ist eine Darstellungsform wie in Abbildung 5.13 möglich, die nicht in einem Fenster, sondern in einem Tab zu sehen ist.

### 5.5.3 Gesamtdarstellung

Wie bei der Entwicklung der Ergebniselement-Darstellung muss auch bei den korrespondierenden Elementen an eine passende Anordnung beziehungsweise Gesamtdarstellung gedacht werden. Denn hierbei handelt es sich letztlich ebenso wie bei den Objekten (und später auch Varianten und Versionen) um nichts anderes als eine Menge von Elementen. Auf welche Art und Weise man diese Art von Menge anordnen und als Gesamtdarstellung anzeigen kann, wurde ausreichend in Abschnitt 5.3 für Objekte analysiert. Deshalb wird

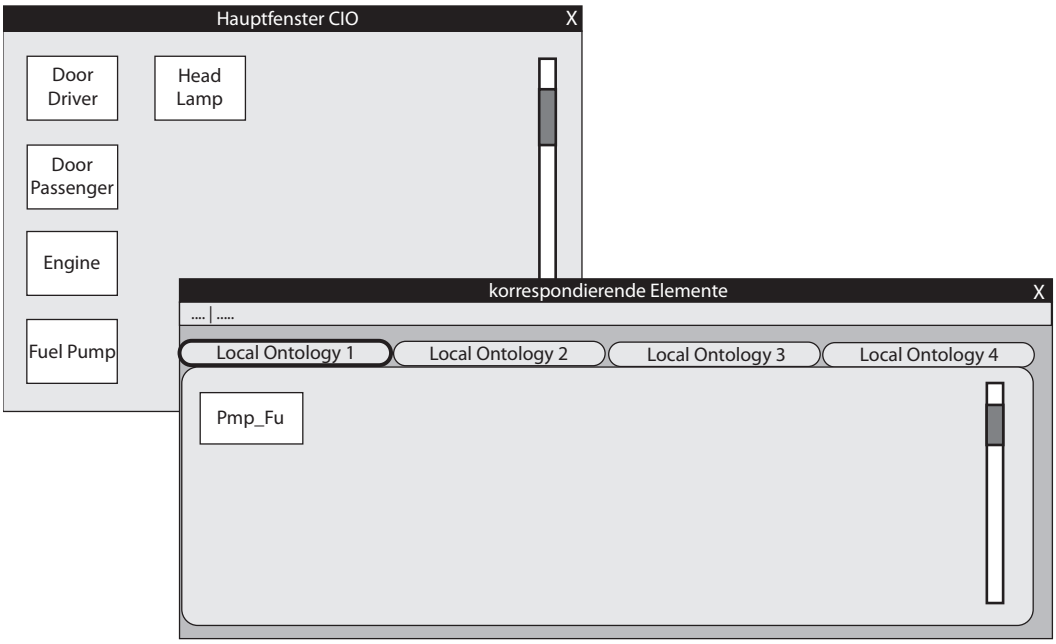


Abbildung 5.12: Ein Fenster mit jeweils einem Tab pro LO wird für die Darstellung der korrespondierenden Elemente verwendet.

in diesem Punkt auf eine nochmalige Analyse verzichtet und auf die Ansätze in diesem Abschnitt verwiesen.

### 5.5.4 Darstellung der Elemente standardmäßig oder auf Wunsch

Die nächste Entscheidung bestimmt, ob die korrespondierenden Elemente (evtl. samt Angabe zu LO) standardmäßig oder erst auf Wunsch des Benutzers angezeigt werden. Hierbei kommt es darauf an, wie hoch man die Bedeutung dieser Information einstuft. Stuft man sie hoch ein, so muss das System diese Elemente zeitgleich zusammen mit der Ergebnismenge anzeigen. Abhängig vom gewählten Darstellungsort stellt das System die korrespondierenden Elemente zusammen mit der Ergebnismenge in einer Ansicht dar oder öffnet automatisch ein eigenes zweites Fenster beziehungsweise einen eigenen zweiten Tab. Stuft man die Bedeutung der korrespondierenden Elemente nicht so hoch ein, reicht es aus, wenn das System eine Schaltfläche anbietet, die dem Benutzer ermöglicht, diese

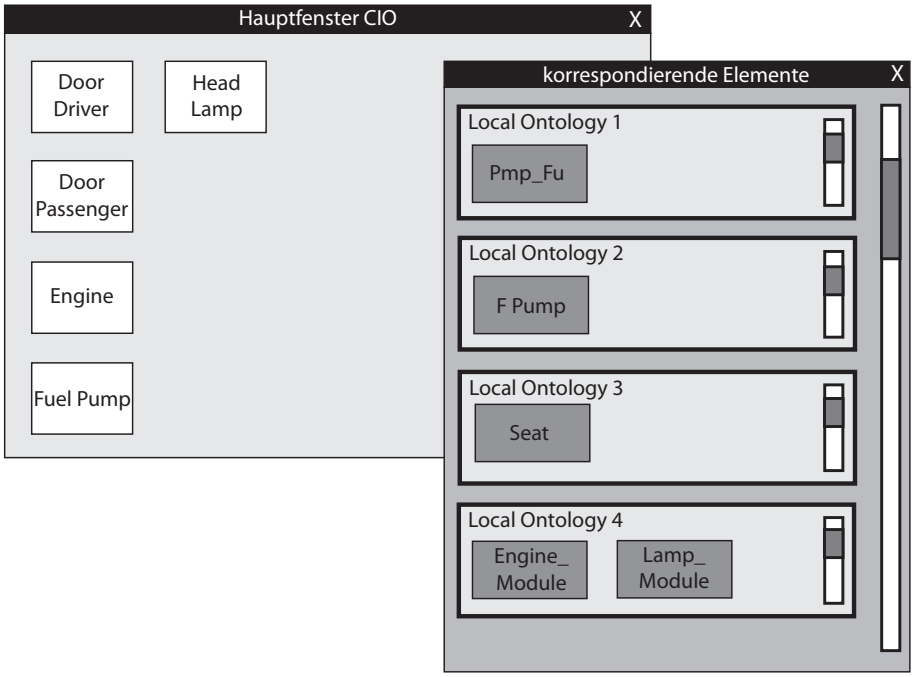


Abbildung 5.13: Ein Fenster mit nur einer Anzeige für die korrespondierenden Elemente, die in den Rahmen der jeweiligen LO positioniert sind.

Elemente auf Wunsch in der Hauptansicht oder in einem neuen Fenster beziehungsweise neuen Tab anzeigen zu lassen. (vgl. Abbildungen 5.14 und 5.15).

### 5.5.5 Zusätzliche Möglichkeiten bei elementweiser Darstellung

Während die bisherigen Ansätze dieses Abschnitts mit einer globalen Anzeige beziehungsweise Interaktion zusammenhängen, ermöglicht die Darstellung auf Wunsch zusätzlich die Alternative der elementweisen Darstellung. Entscheidet man sich für die elementweise Anzeige der korrespondierenden Elemente, ergibt sich dadurch eine wesentlich übersichtlichere und von Benutzerseite aus einfach handhabbare Arbeitsumgebung. Auf Klick eines CIO-Elements oder durch eine ähnliche Interaktion damit kann sich der Benutzer für dieses spezielle Element die dazugehörigen LO-Elemente anzeigen lassen (z.B. mit Hilfe eines Tooltips (vgl. Abbildung 5.9), eines Dialogs oder eines kleinen Fensters (vgl. Abbildungen 5.16 und 5.17)). Hat der *Business User* alle nötigen Informationen aus dieser Anzeige

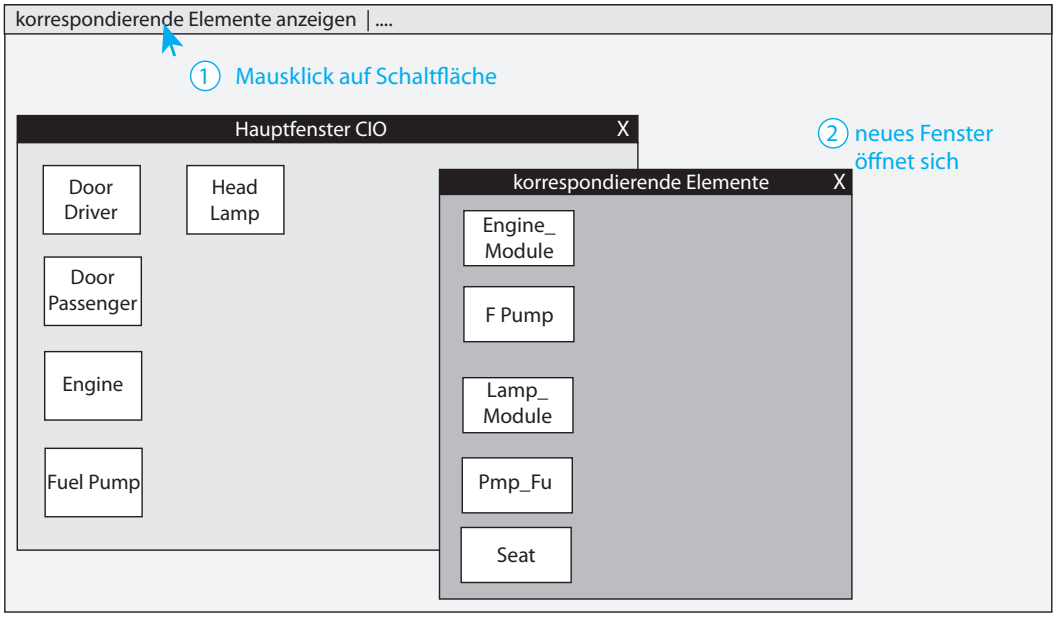


Abbildung 5.14: Auslagerung in ein neues Fenster nach Klick auf entsprechende Schaltfläche.

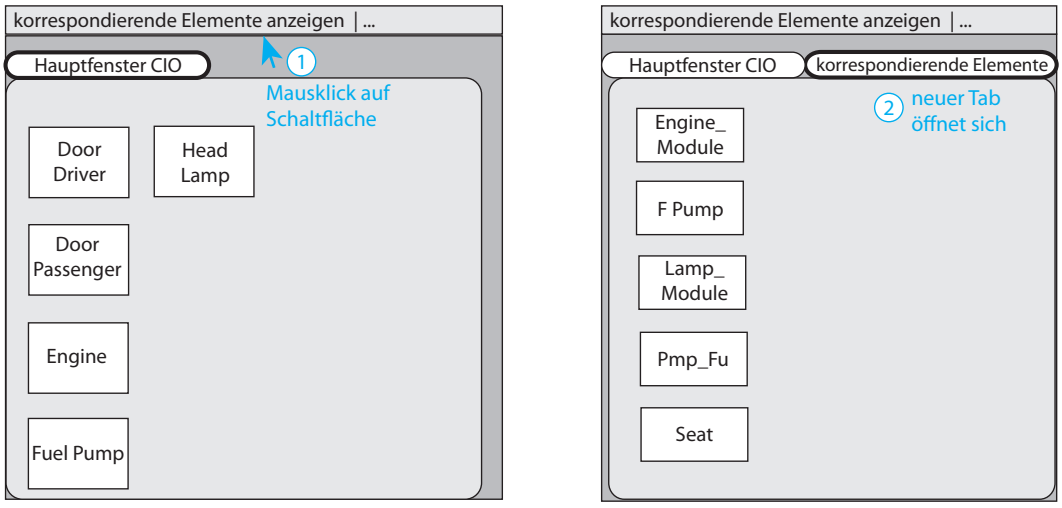


Abbildung 5.15: Auslagerung in ein neues Tab nach Klick auf entsprechende Schaltfläche.

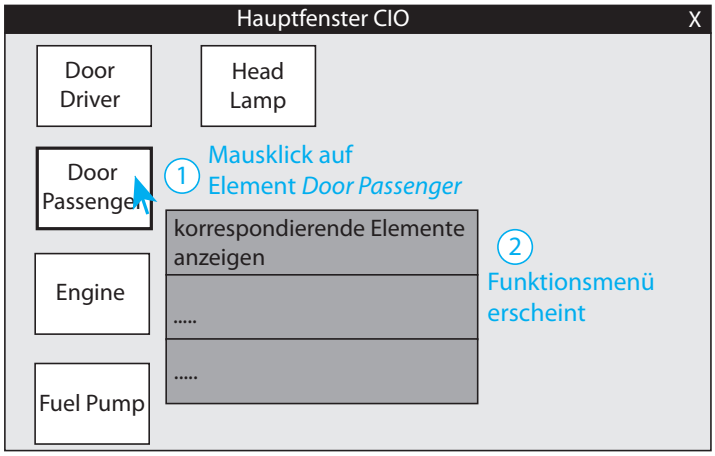


Abbildung 5.16: Durch Klicken eines Elements erscheint das Funktionsmenü mit der Schaltfläche „tiefere Elementebenen anzeigen“.

entnommen, kann diese offen bleiben, minimiert oder wieder geschlossen werden. Ob mehrere solcher einzelner Ansichten erlaubt sind, hängt vom jeweiligen endgültigen Ansatz ab. Alternativ könnte sich beim Klick auf ein anderes Element die alte Anzeige schließen und die Neue öffnen.

Im Gegensatz dazu ermöglicht eine globale Interaktion wie in Abschnitt 5.5.4 die Informationsanzeige aller korrespondierenden Elemente. Dadurch muss sich der Benutzer nicht erst durch die möglicherweise hunderte Elemente einzeln durchklicken, um sich über die jeweiligen korrespondierenden Elemente zu informieren. Andererseits wird bei diesem Ansatz eine Zuordnung zwischen Ergebniselement(en) und korrespondierenden Elementen (siehe Abschnitt 5.5.6) eventuell schwieriger. Es sprechen somit einige Argumente für eine globale Interaktion und Darstellung, aber gleichzeitig auch viele Argumente für eine elementweise. Letztlich ist keiner der beiden Ansätze perfekt und ohne Nachteile, weshalb es letztlich auf eine Abschätzung ankommt, was für den Benutzer am besten ist.

### 5.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen darstellen

Trotz aller bisherigen Überlegungen ist noch völlig ungeklärt, wie man die Zusammengehörigkeit der CIO-Elemente und deren korrespondierender Elemente der LOs darstellt.

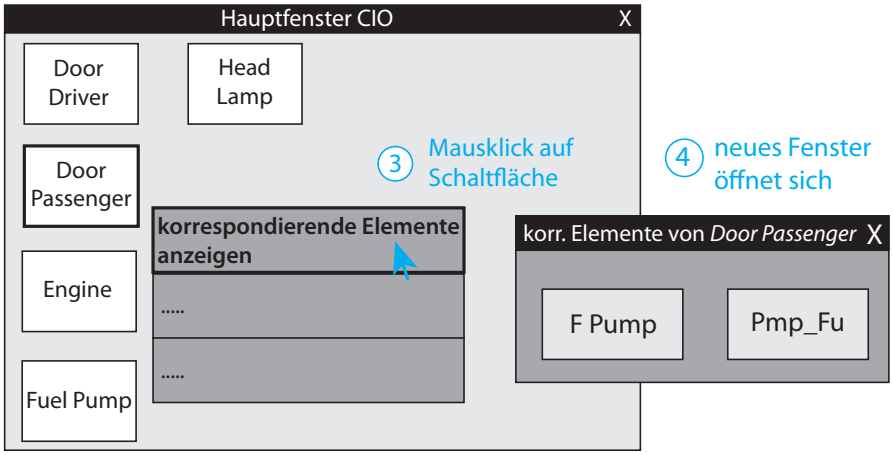


Abbildung 5.17: Klickt der Benutzer auf die Schaltfläche, erscheint ein neues Fenster mit dem Namen des Elements benannt und den korrespondierenden Elementen darin.

Bis zu diesem Punkt wurde zwar über den Darstellungsort, die nötigen Informationen und die Art und Weise der Interaktion und Darstellung gesprochen, jedoch lässt keine der gezeigten Abbildungen eine richtige Zusammengehörigkeit zwischen korrespondieren und CIO-Elementen erkennen. Die einfache räumliche Nähe der Elemente wie in Abbildung 5.10 beispielsweise reicht noch nicht aus, um definitiv aussagen zu können, welche korrespondierenden Elemente zu welchen Elementen der Local Ontologies gehören. Um die Zusammengehörigkeit zu verdeutlichen, kann man zum Beispiel auf direkte Verknüpfungen (z.B. durch Linien) oder auf farbliche Unterlegungen zurückgreifen (vgl. Abbildung 5.18).

Abbildung 5.18 zeigt ein Beispielergebnis einer Datenabfrage. In diesem Fall wurde der Einfachheit halber eine gemeinsame Darstellung in einer Ansicht gewählt und die korrespondierenden Elemente am Rand der Anzeige werden nur durch ihren Namen repräsentiert. Der entscheidende Unterschied zwischen der linken und rechten Ansicht ist die Art der Verknüpfung: Links wurde eine direkte Verknüpfung durch Linien gewählt und rechts ein farblicher Zusammenhang hergestellt. Sind die Elemente nicht gemeinsam in einer Anzeige platziert, so sind die korrespondierenden Elemente nach den Erkenntnissen der vorherigen Abschnitte entweder in einem extra Fenster oder in einem neuen Tab ausgelagert. Verknüpfungen machen in beiden Fällen weniger Sinn, denn Verknüpfungen zwischen zwei Fenstern/Tabs können darstellungstechnisch nur schwer zugeordnet werden. Farbliche Unterlegungen kommen in diesem Kontext deutlich besser zum Tragen.

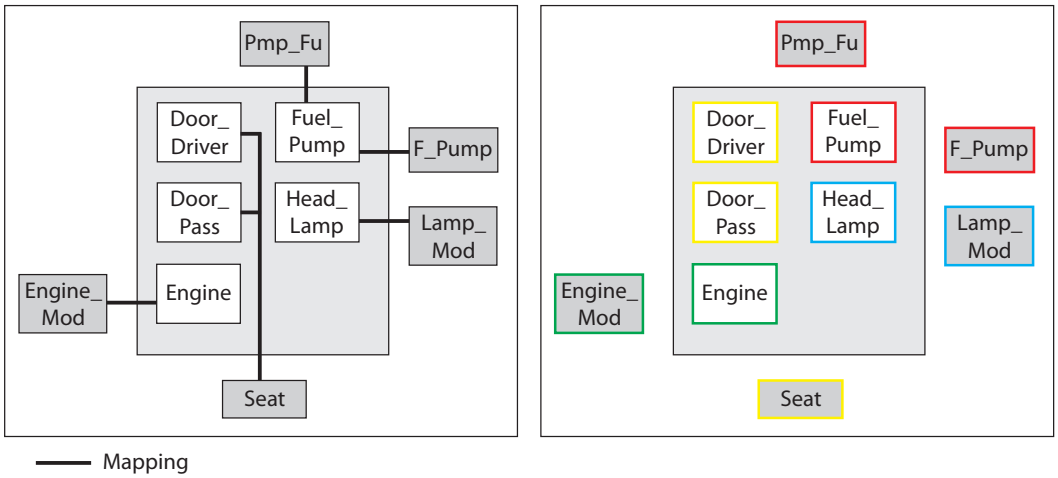


Abbildung 5.18: Beispiele verschiedener Darstellungsarten von Zusammengehörigkeit: direkte Verknüpfungen (links) und Verwendung von Farbe (rechts).

Der Inhalt zweier Fenster, die sich der Benutzer auf dem Bildschirm aneinander platziert, lässt sich gut analysieren und vergleichen (vgl. Abbildung 5.19). Die farbliche Zusammengehörigkeit fällt dem Benutzer dabei direkt ins Auge. Andererseits sinkt die Unterscheidbarkeit der farblichen Elemente mit steigender Anzahl an Ergebniselementen. Außerdem ist das Problem bei der Verwendung von Tabs, dass jeweils nur ein Tab angezeigt werden kann. Wenn man als Benutzer erst zwischen beiden Tabs hin und her klicken muss, ist der kognitive Aufwand wesentlich höher. Grund dafür ist, dass sich der Benutzer ein CIO-Element aussuchen muss, sich dessen Farbe merken muss, daraufhin zum Tab der korrespondierenden Elemente klicken muss und dort die farblich passenden Elemente finden muss. Um dem Benutzer diese eigentlich einfache Aufgabe nicht unnötig zu erschweren, sollte man bei der Entwicklung entweder auf eine Tab-Anzeige verzichten oder den Benutzer so unterstützen, dass sich dieser Tab automatisch in ein eigenes Fenster umwandelt. Dadurch kann der Benutzer die beiden Fenster wieder nebeneinander am Bildschirm platzieren und wesentlich leichter seiner Arbeit nachgehen. Andernfalls muss dem Benutzer bewusst sein, dass er den Tab als Fenster auf dem Bildschirm anordnen kann.

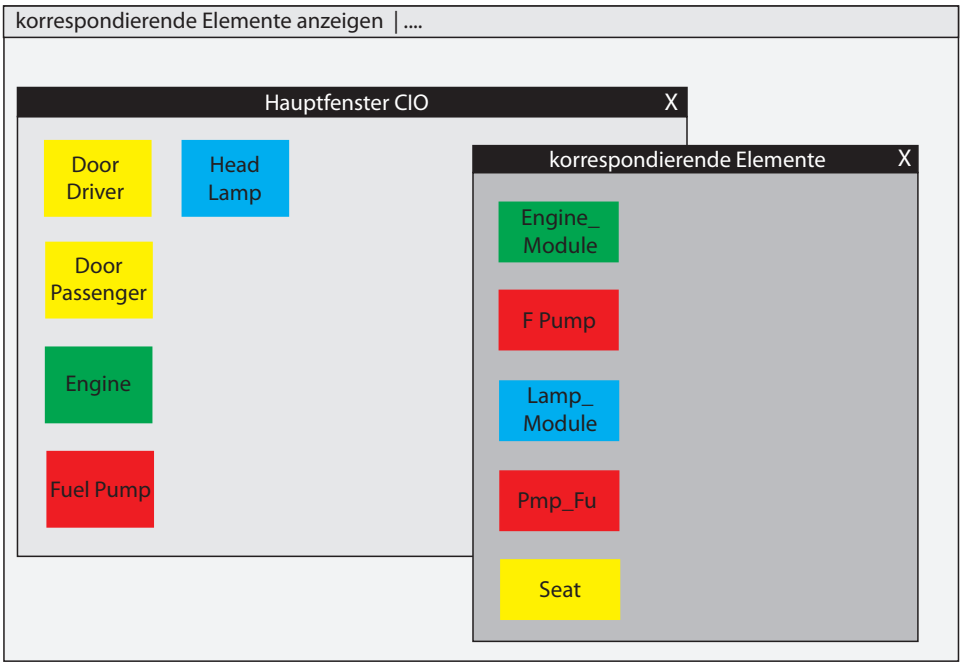


Abbildung 5.19: Farbliche Zusammengehörigkeit bei Verwendung eines Extra-Fensters.

### 5.5.7 Berücksichtigung von mehrwertigen Korrespondenzen

In Kapitel 2 dieser Arbeit wurden verschiedene Korrespondenzen beschrieben, die zwischen Schema-Concepts, Individuals und Attributen bestehen können. Gilt dabei eine 1:1 Korrespondenz, so wird jedes korrespondierende Element aus den LOs im entsprechenden Bereich dargestellt. Gilt jedoch eine 1:n Korrespondenz (z.B. Subsumes), so gibt es für ein Element aus einer Local Ontology mehrere korrespondierende CIO-Elemente. Die Frage die sich hierbei stellt ist, ob dieses Element dann jeweils für jedes der *n* CIO-Elemente dargestellt wird oder ob es nur einmal dargestellt wird. Je nach Kenntlichmachung der Verbindung (Verknüpfung, Farbe) macht eines der beiden Ansätze mehr Sinn. Bei Verknüpfungen durch Linien beispielsweise müsste das Element der Local Ontology so positioniert werden, dass die Linien der *n* korrespondierenden CIO-Elemente so übersichtlich wie möglich gezogen werden können. Bei einer farblichen Zusammengehörigkeit reicht es wiederum, wenn man das korrespondierende Element nur einmal darstellt. Der Zusammenhang ergibt sich dann aus den mehreren farblich hinterlegten CIO-Elementen. Die Wahl eines dieser genannten Möglichkeiten hängt stark von mehreren Faktoren wie der Ergebnismenge, der Darstel-

lungsform, dem Darstellungsrahmen, dem verfügbaren Platz und der Menge von LOs sowie korrespondierenden CIO-Elementen ab. Bei der Verwendung von Linien oder anderen Verknüpfungen kommt es stark auf die Anordnung der Elemente an. Je mehr Elemente und je unvorteilhafter diese abgebildet sind, desto eher überschneiden sich die Verknüpfungen und werden von anderen Elementen überdeckt. Auch das farbliche Repertoire ist eher nur bei kleineren Ergebnismengen sinnvoll einsetzbar. Während sich bei sechs Elementen deutlich unterscheidbare Farben finden lassen, so ist dies bei höheren zweistelligen Ergebnismengen nicht mehr der Fall. Eine ebenfalls an dieser Stelle wichtige, zu erwähnende Abhängigkeit dieser Entscheidung ist der Darstellungsort der korrespondierenden Elemente. Eine Verknüpfung macht nur dann Sinn, wenn sich die Elemente auf einer Bildebene befinden, d.h. in einem Fenster und auf einem gemeinsamen Tab sind. Sind die CIO-Elemente z.B. in Fenster A und die korrespondierenden Elemente in Fenster B oder gar in verschiedene Tabs aufgeteilt, so machen Verknüpfungen durch Linien beispielsweise keinen Sinn. In diesem konkreten Beispiel ist, wenn überhaupt, eine farbliche Unterlegung passender. Dabei ist gerade bei mehreren Elementen zu überlegen, ob eine farbliche Zusammengehörigkeit nur beim Klick auf ein CIO-Element erfolgt. Dies würde dem oben genannten Problem der Farbenvielfalt entgegenwirken. Gerade an dieser Stelle sind Erweiterungen der eigentlichen Idee denkbar.

## 5.6 Anforderung 6: Ebenenwechsel zu Varianten und Versionen

Ein letzter zentraler und wichtiger Punkt des gesamten PROCEED-Frameworks sind die verschiedenen Datenebenen Product Data Collection, Object, Variant und Version. Bekannt ist, dass die Datenabfragen gewöhnlich auf Objektebene beginnen. Davon ausgehend soll jedoch ein Ebenenwechsel vollzogen werden können. Das bedeutet, dass das System sowohl die Objekte darstellen als auch in der Lage sein muss, gegebenenfalls neue Anzeigen für Varianten und Versionen einzelner und/oder aller Objekte zu erstellen. Dieser Vorgang soll durch entsprechende Interaktion vom Benutzer vollzogen und die Elemente durch passende Navigation angesteuert werden können. Wie schon bei der Betrachtung der Objekte können auch hier die in Kapitel 4 analysierten Anforderungen abgeleitet und als Grundlage für die Gestaltung betrachtet werden.

### 5.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten

Varianten und Versionen bestehen wie Objekte aus einem Namen und dazugehörigen Attributwerten. Wie schon bei den Objekten, sollten Varianten und Versionen nur so viel wie nötig, aber so wenig wie möglich Informationen anzeigen müssen. Deshalb wird es am geeignetsten sein, sich auch hier auf den Namen des Elements samt einem äußeren Rahmen zu beschränken. Zur besseren Unterscheidbarkeit und Übersichtlichkeit sollte die Wahl eines andersförmigen Rahmens für jeweils die Objekten, Varianten und Versionen gewählt werden. So könnten beispielsweise die Objekte rechteckig, die Varianten kreisförmig und die Versionen mehreckig sein (vgl. Abbildung 5.20). Bezüglich der Attributliste der Varianten und Versionen gilt das gleiche wie für die der Objekte. Auch hier muss man sich bei der endgültigen Entwicklung eines visuellen Ansatzes entscheiden, ob man die Attribute standardmäßig oder erst nach Benutzerinteraktion anzeigen möchte. Wie dies dann im konkreten Fall aussehen kann und welche Möglichkeiten der Darstellung es dabei gibt, ist ausführlich in Abschnitt 5.2 beschrieben.

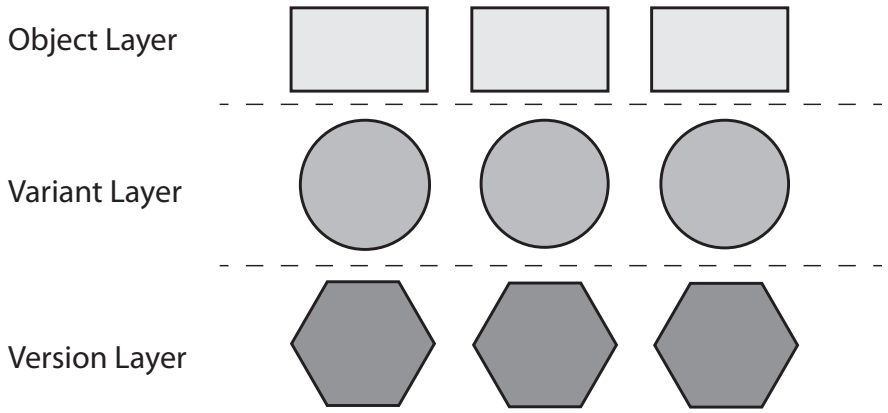


Abbildung 5.20: Verschiedene Formen für Objekte, Varianten und Versionen.

### 5.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen

Je tiefer sich der Benutzer durch die Datenebenen navigiert, auf desto mehr Elemente wird er für gewöhnlich stoßen. Somit werden die in Abschnitt 5.3 analysierten Gesamtdarstellungs-

Möglichkeiten für Varianten und vor allem Versionen von noch wichtigerer Bedeutung. Wie aus diesem Abschnitt bekannt ist, kommen beispielsweise mehrspaltige Listen, eine geometrische Darstellung als Rechteck oder Kreis oder eine eher unübersichtlichere ungeordnete Darstellung in Frage. Bei Gestaltungsfragen ähnlicher Komponenten kommt es darauf an, ob man als Entwickler das Kriterium der Konsistenz schwächer oder stärker berücksichtigen will. Entscheidet man sich bei der Darstellung der Objekte beispielsweise für eine mehrspaltige Listendarstellung, so stellt sich die Frage, ob man diese für die Varianten und Versionen ebenfalls umsetzen soll oder sich für eine unterscheidbare Darstellung entscheidet. Während die Wahl der gleichen Gesamtdarstellung dem System einen gestaltungstechnisch konsistenten Eindruck verleiht, sind anderweitige Darstellungen der Varianten und Versionen aufgrund ihrer Vielzahl möglicherweise besser geeignet. An dieser Stelle gilt es also beim endgültigen Ansatz zu begründen, wieso man sich gerade für Gesamtdarstellung X (im Vergleich zu Y) entschieden hat.

### 5.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen

Um die Varianten und Versionen bestmöglich für den Benutzer am Bildschirm platzieren zu können, ist erneut die Frage des geeigneten Rahmens für die Gesamtdarstellung zu stellen. Wie schon in den beiden vorherigen Unterkapiteln kann auch in diesem auf einen früheren Abschnitt (Abschnitt 5.4) für die möglichen Umsetzungen eines Rahmens verwiesen werden. Jedoch ist an diesem Punkt eine gemeinsame Darstellung mit den Objekten der Ergebnismenge und möglicherweise auch noch deren korrespondierender Elemente allein aus Platzgründen kategorisch auszuschließen. Nichtsdestotrotz liefert die Wahl von Fenstern oder Tabs beispielsweise genug Möglichkeiten, um dem Benutzer die nötigen Informationen auf eine wesentlich angenehmere und aufgabenfreundlichere Art und Weise darzulegen.

#### Fenster

Entscheidet man sich bei einem Ebenenwechsel zu Varianten und Versionen für Fenster als Darstellungsrahmen (vgl. Abbildung 5.21), konfrontiert man den Benutzer an dieser Stelle erneut mit den Vor- und Nachteilen von Fenstern.

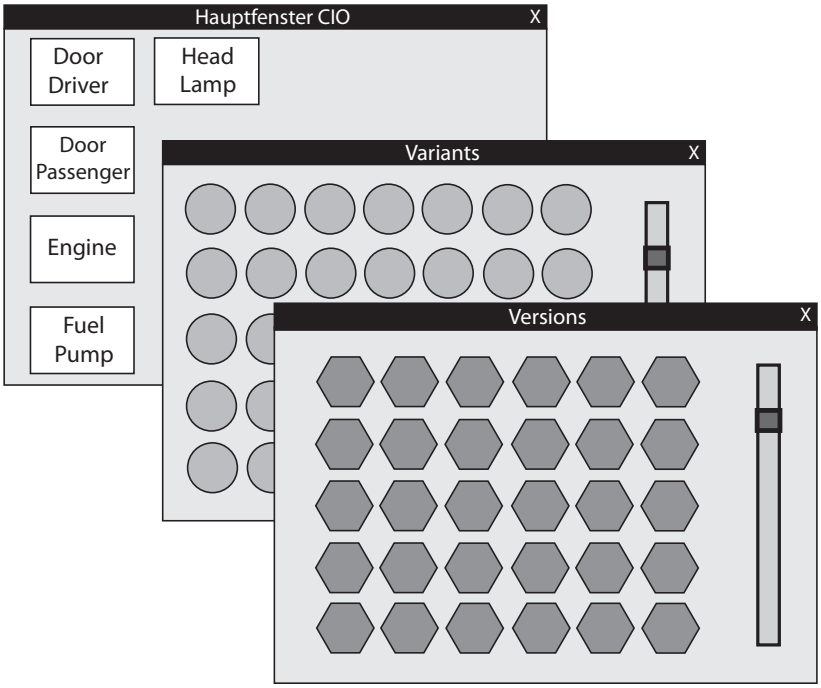


Abbildung 5.21: Für jede Datenebene (Object, Variant, Version) wird ein eigenes Fenster verwendet.

Es erscheint zwar ein neues Fenster, das seinen Platz entweder auf der Taskleiste oder am Bildschirm benötigt, aber dieses bietet dem Benutzer gleichzeitig einen schnellen und unkomplizierten Zugriff samt einfacher Vergleichsmöglichkeit mit anderen Anzeigen. Dieses neue Fenster bietet ausreichend Platz für entsprechende Darstellungen der Elemente der tieferen Ebene und ist mit nur einem Mausklick von der Bildfläche wieder verschwunden. Alternativ kann in einem gemeinsamen Fenster für die zwei tieferen Ebenen jeweils ein Tab angeboten werden (vgl. Abbildung 5.22).

**Tabs**

Auch die Wahl von einem neuen Tab als Darstellungsfläche einer neuen Ebene erweist sich als einfach und praktisch für diesen Zweck. Durch einen vom Benutzer interagierten Ebenenwechsel erscheint ein neuer Tab in der Leiste, der für die Varianten oder Versionen

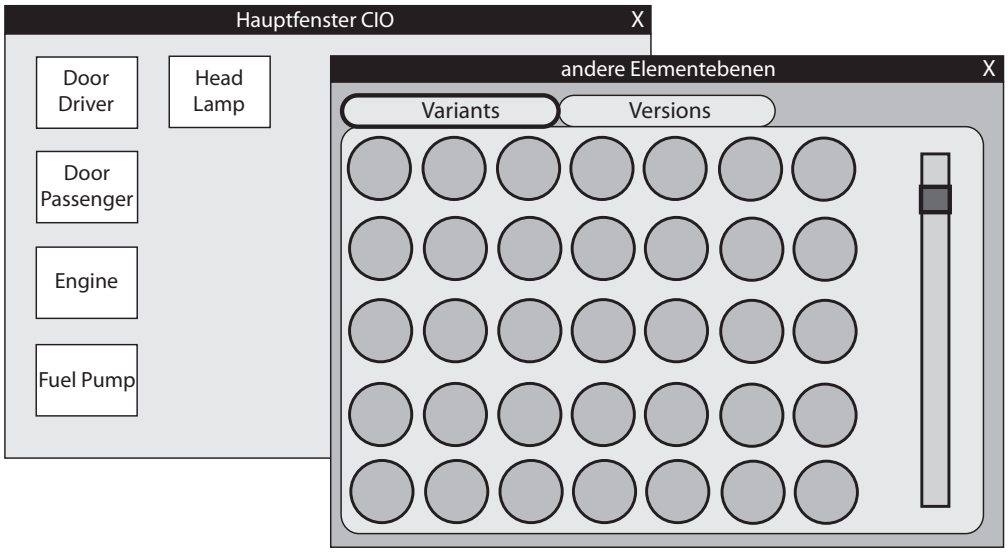


Abbildung 5.22: Statt mehrerer Fenster wird nur eines verwendet. Dieses beinhaltet zwei Tabs für die beiden tieferen Elementebenen.

als Darstellungsplatz (evtl. mit Hilfe von Unterstützungswerkzeugen wie Scrollbars oder einer Schaltfläche zum Umblättern einer Anzeigenseite) ausreicht (vgl. Abbildung 5.23).

**Seitenansicht**

Statt einen der beiden oberen Ansätze zu nehmen, kann eine weitere Idee, die Anwendung mehrerer Seiten auf einer einzigen Bildschirmfläche, nützlich sein. Statt einigen Fenstern oder Tabs wird die Darstellung eines Ergebnisses einer Datenabfrage wie eine Art Buch gehandhabt. Dies bedeutet, dass man als Benutzer auf den ersten Seiten beispielsweise die Objekte des Ergebnisses dargestellt bekommt, auf den daran anschließenden Seiten die Varianten all dieser Ergebnisobjekte und zum Schluss des kleinen „Buches“ findet der Benutzer alle Versionen der Abfrage (vgl. Abbildung 5.24). Inwiefern sich dieser Ansatz mit weiteren Anforderungen (z.B. den korrespondierenden Elementen) vereinbaren lässt, kommt auf den jeweiligen endgültigen Ansatz an.

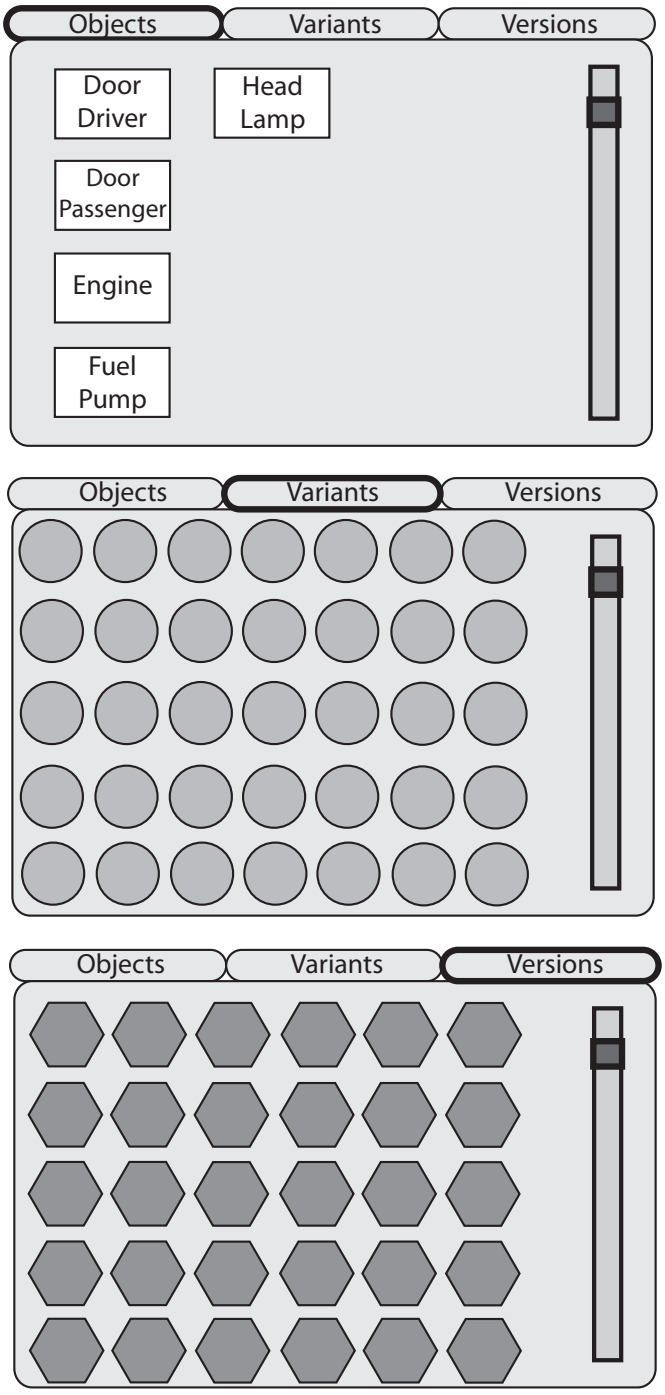


Abbildung 5.23: Verwendung von einzelnen Tabs pro Elementebene.

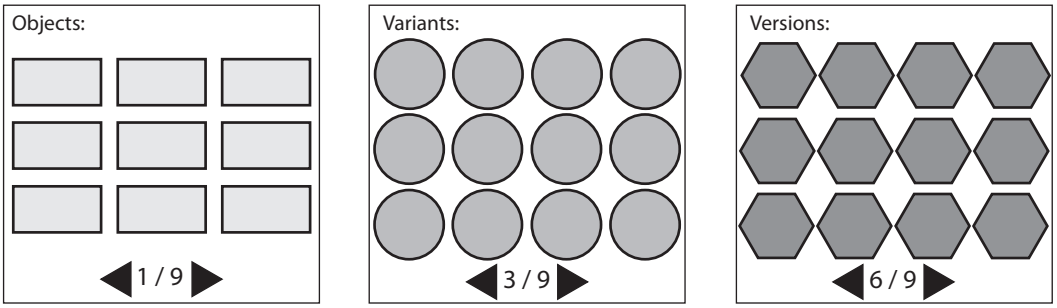


Abbildung 5.24: Seitenansicht der verschiedenen Elementebenen. Das „Buch“ beginnt mit den Objekten, gefolgt von Varianten und den Versionen am Ende.

### 5.6.4 Ebenenwechsel - global oder elementweise

Da nun einige Darstellungsmöglichkeiten und -rahmen für verschiedene Ebenen feststehen, muss noch analysiert werden, auf welche Art und Weise der Ebenenwechsel vollzogen werden soll. Wie schon wiederholt in anderen Anforderungen angesprochen, kommt auch hier erneut die Unterscheidung zwischen global und elementweise ins Spiel. Dabei können die vorherig genannten Darstellungsrahmen sowohl für eine globale als auch für eine elementweise Herangehensweise verwendet werden.

#### Globaler Ebenenwechsel

Die eine Möglichkeit ist, den Ebenenwechsel global mit einer entsprechenden Schaltfläche anzubieten (vgl. Abbildung 5.25). Diese Schaltfläche sollte entweder global innerhalb des systeminternen Rahmens oder in den beiden analysierten Darstellungsrahmen für Ergebnis- und korrespondierende Elemente zu finden sein. Klickt man beispielsweise auf eine ähnliche Schaltfläche wie in Abbildung 5.25 dargestellt, so muss einer der in Abschnitt 5.6.3 genannten Darstellungsrahmen (Fenster, Tabs, Seitenansicht) durch das System sichtbar werden. Dieser beinhaltet dann alle Varianten oder Versionen (je nach Ebenenwechsel), die an den Ergebnisobjekten dranhängen. Diese Möglichkeit sollte zum Einsatz kommen, wenn man es als Entwickler für wichtiger hält, dem Benutzer schnell und einfach alle die für seine Abfrage betroffenen Elemente anzuzeigen. Dadurch erhält der Benutzer jedoch bei größeren Abfragen eine enorme Vielzahl an Elementen auf verschiedenen Ebenen, ohne jeglichen Bezug auf die vorherige Ebene. Er weiß somit nicht, welche Variante zu

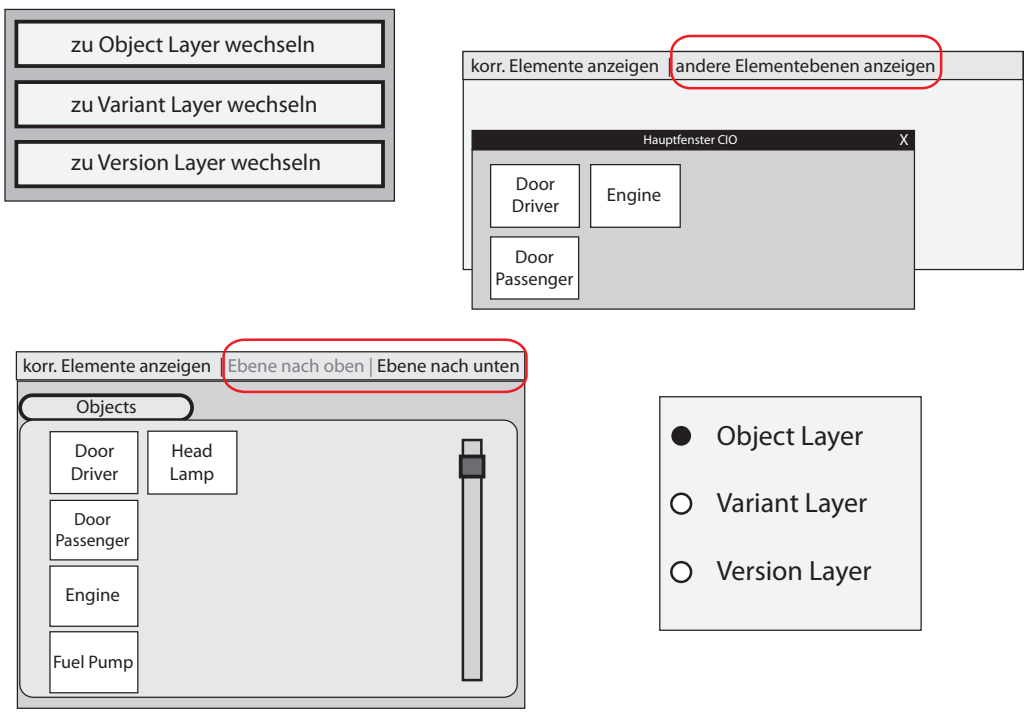


Abbildung 5.25: Verschiedene Möglichkeiten der Darstellung einer globalen Schaltfläche für die Interaktion in eine andere Ebene.

Objekt A aus der vorherigen Ansicht gehört und umgekehrt (vgl. Abbildung 5.26). Dieser fehlende Zusammenhang muss entweder in Kauf genommen oder erst noch passend dazu entwickelt werden. Nichtsdestotrotz kann eine umfassende Entwicklungsarbeit eine zusätzliche Funktion ermöglichen, die dem Benutzer sowohl einen globalen als auch einen elementweisen Ebenenwechsel anbietet. Dadurch erhielte der Benutzer die volle Kontrolle über die Handlungsschritte und könnte selbständig entscheiden, welche Information ihm im konkreten Fall wichtiger ist. Wie genau ein elementweiser Ebenenwechsel aussehen kann, beschreibt der folgende Abschnitt.

#### Elementweiser Ebenenwechsel

Die andere Möglichkeit eines Ebenenwechsels ist die, dass der Benutzer sich individuell durch nur einzelne Objekte in dessen tiefere Ebenen navigieren kann. Diese Art der Navi-

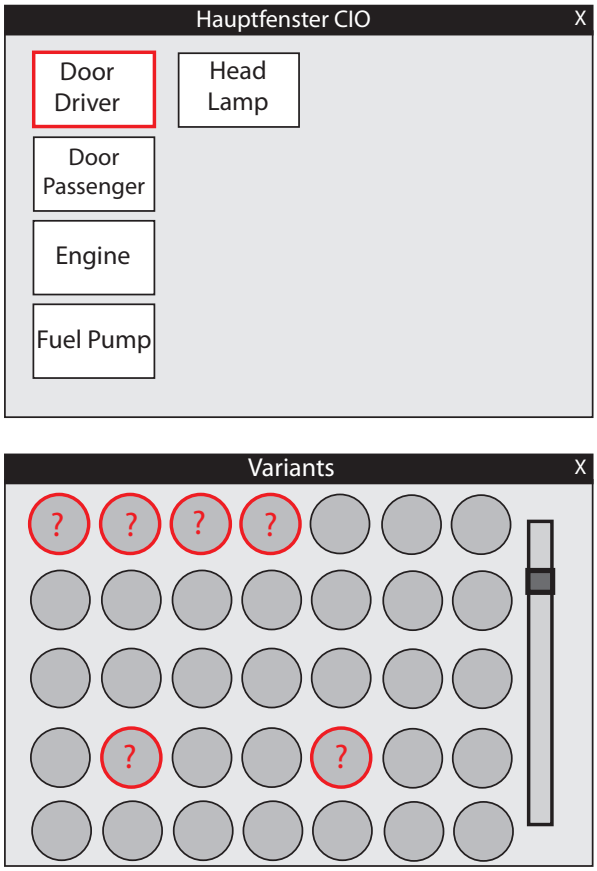


Abbildung 5.26: Nach einem globalen Ebenenwechsel kann nicht zugeordnet werden, wie die Elemente der unterschiedlichen Ebenen in Beziehung zueinander stehen.

gation lässt sich durch Klick auf ein Element sowohl mit einer globalen Schaltfläche (vgl. Abbildung 5.27) als auch mit einer elementweisen Schaltfläche umsetzen (vgl. Abbildung 5.28). Wie schon in Abschnitt 5.5.5 bei der elementweisen Darstellung der korrespondierenden Elemente, kann man auch hier die Anzeige der Elemente der tieferen Ebene zum Beispiel in einem neuen, kleinen Fenster platzieren (vgl. Abbildung 5.29). Alternativ kann beispielsweise auf die Tab-Darstellung wie in Abbildung 5.30 zurückgegriffen werden.

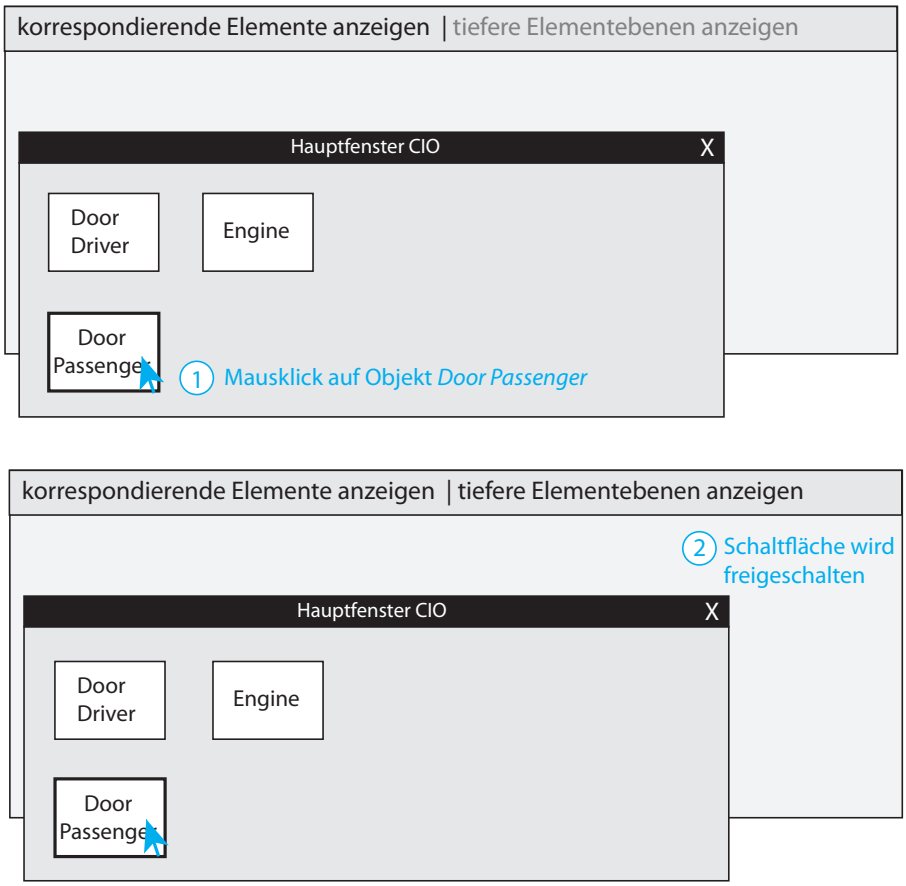


Abbildung 5.27: Ein angeklicktes Element aktiviert die globale Schaltfläche für den elementweisen Ebenenwechsel.

### 5.6.5 Ebenen der korrespondierenden Elemente berücksichtigen

Je nach Entscheidung für einen globalen oder elementweisen Ebenenwechsel müssen noch zusätzliche Überlegungen hinsichtlich der korrespondierenden Elemente gemacht werden. Denn korrespondierende Elemente sind ebenfalls entweder Objekte, Varianten oder Versionen. Entscheidet man sich für einen elementweisen Ebenenwechsel bei den Ergebniselementen, so können alle dafür analysierten Darstellungen und Eigenschaften auch für die korrespondierenden Elemente übernommen werden. Einzig beim Fensternamen kann man überlegen, ob eine Zusatzkennung für das korrespondierende Element sinnvoll ist. Bei einem globalen Ebenenwechsel kommen je nach Darstellungsort der korrespondierenden

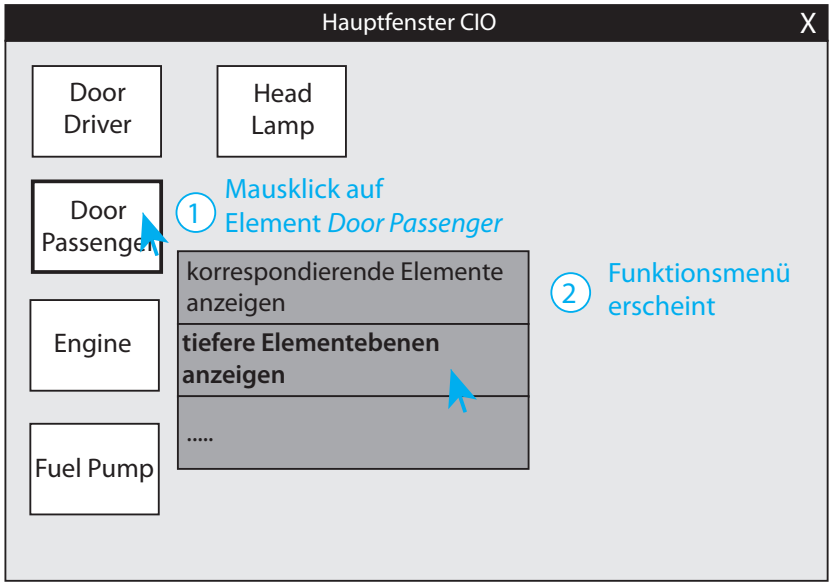


Abbildung 5.28: Durch Klicken auf ein Element erscheint ein Funktionsmenü am Mauszeiger, der dem Benutzer u.a. erlaubt, die tieferen Ebenen anzuzeigen.

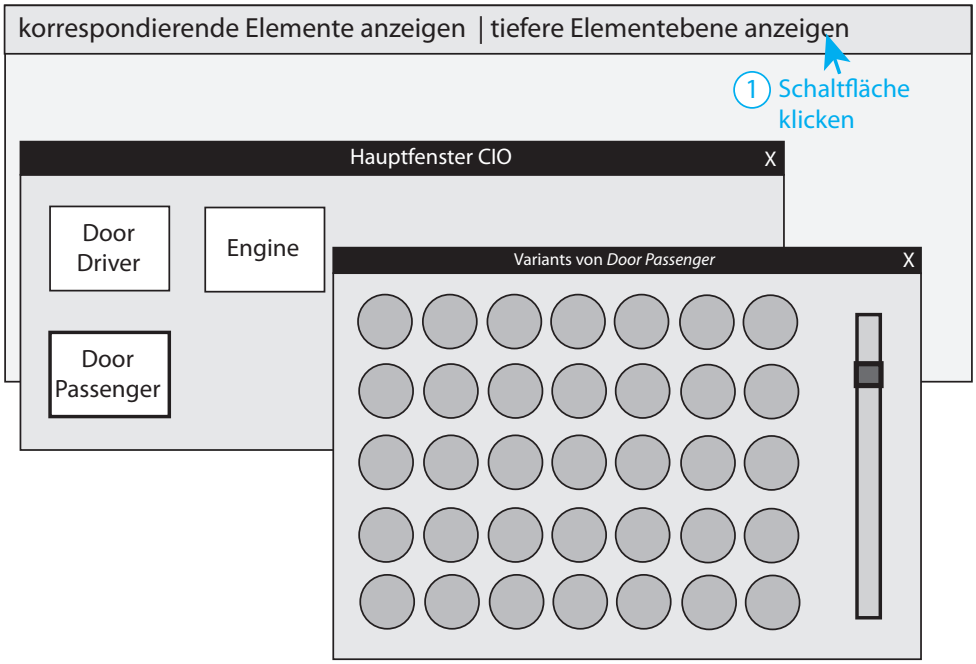


Abbildung 5.29: Verwendung eines eigenen Fensters für die Variantendarstellung eines einzelnen Objekts (hier: Door Passenger).

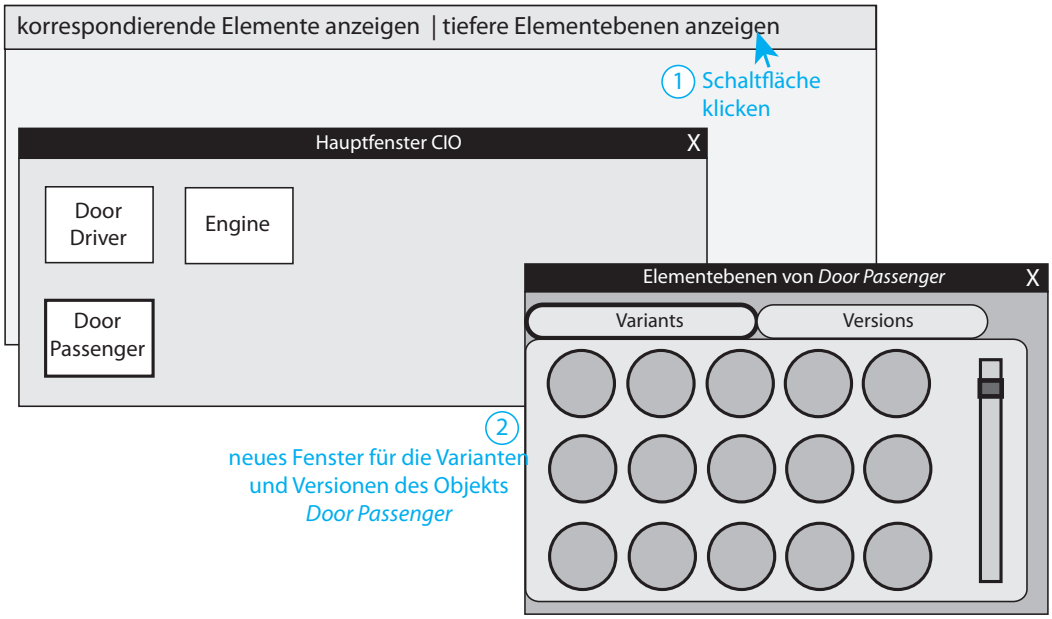


Abbildung 5.30: Alternative Darstellung der beiden tieferen Datenebenen eines einzelnen Elements in zwei einzelnen Tabs.

Elemente entsprechende Anforderungen beziehungsweise Möglichkeiten zur Umsetzung ähnlich in Frage:

**Fall 1**: Die korrespondierenden Objekte werden gemeinsam mit den Ergebnisobjekten in einer Ansicht dargestellt. Klickt der Benutzer nun auf die Schaltfläche für einen Ebenenwechsel, so sollten hierbei auch die korrespondierenden Objekte durch die passenden Varianten beziehungsweise Versionen ausgetauscht werden. Da dieser Vorgang bei Wahl der gemeinsamen Darstellung zu viel Platz beansprucht, wird davon für die Wahl des Darstellungsrahmens der korrespondierenden Elemente erneut abgeraten.

**Fall 2**: Die korrespondierenden Objekte werden in einem eigenen Fenster angezeigt. Um zwischen korrespondierenden Objekten, Varianten und Versionen zu unterscheiden, können sowohl einzelne Fenster (vgl. Abbildung 5.21), die in Abschnitt 5.4.3 und 5.6.3 erläuterte Seitenansicht (vgl. Abbildung 5.24) als auch Tabs verwendet werden (vgl. Abbildung 5.31). Jeder Tab entspricht dabei einer Ebene und zeigt die entsprechenden Elemente aus den verschiedenen Local Ontologies an. Für Vergleichszwecke kann sich der Benutzer hierbei wieder insofern bereichern, als dass er einen Tab als einzelnes Fenster umfunktionieren

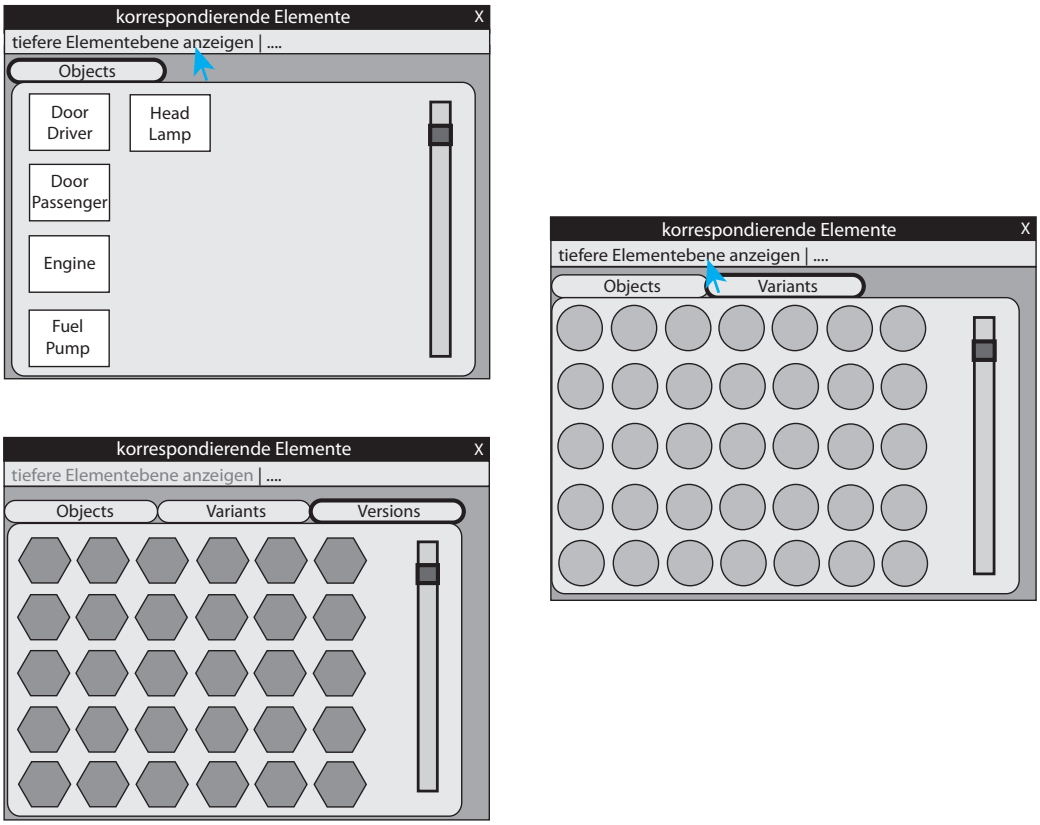


Abbildung 5.31: Darstellung der korrespondierenden Elemente in einem eigenen Fenster mit drei eigenen Tabs.

kann. Verwendet man verschiedene Seiten, so entspricht dies wieder einer Art Buch, in welchem Seite für Seite erst die Objekte, dann die Varianten und zuletzt die korrespondierenden Versionen erscheinen. Alternativ zu eigenen Tabs und Seiten könnte pro Ebene ein eigenes Fenster aufgehen, welches die für die ausgewählte Ebene entsprechenden Elemente beinhaltet. Da sich die Tiefe der Datenebenen auf höchstens vier beschränkt, ist die Anzahl der geöffneten Fenster noch im Rahmen des ermesslichen.

**Fall 3**: Die korrespondierenden Objekte werden neben dem CIO-Tab in einem eigenen Tab angezeigt. Dann ist es möglich, für jede Ebene einen weiteren Tab zu öffnen, sobald der Benutzer mit dem System entsprechend interagiert (vgl. Abbildung 5.32). Andernfalls ist auch hier eine Seitenregelung möglich. Dies erspart das Erscheinen weiterer Tabs, auf das der Benutzer nicht unbedingt vorbereitet ist. Nachteilig ist in diesem Falle jedoch das

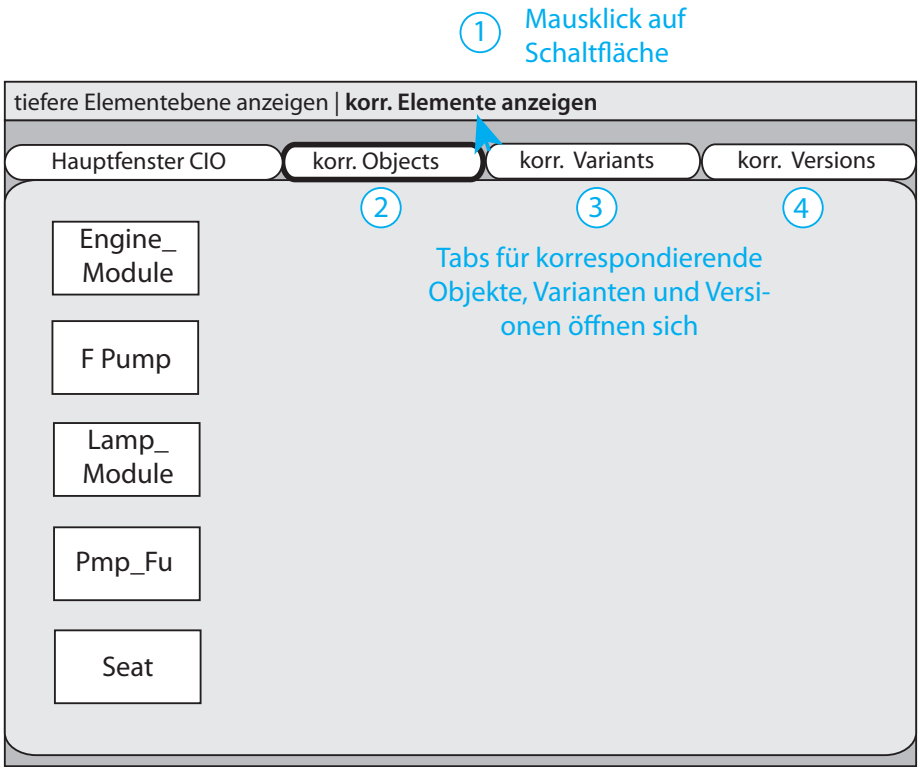


Abbildung 5.32: Verwendung einzelner Tabs für korrespondierende Objekte, Varianten und Versionen, falls Tabs als allgemeiner Darstellungsrahmen gewählt wurden.

Durchklicken der Seiten, denn bei der Vielzahl der Elemente kann es sich um ziemlich viele Seiten handeln. Da der Benutzer aber recht schnell einen Überblick haben möchte, muss abgewogen werden, welche Nachteile überwiegen.

In diesem Kapitel sind nun alle aus Kapitel 4 gestellten Anforderungen auf die Gestaltung übertragen und erste Entwürfe und Ideen für eine visuelle Umsetzung entwickelt worden. Darauf aufbauend können somit in den beiden nächsten Kapiteln 6 und 7 komplette Ansätze für die Visualisierung und Navigation komplexer Produktdaten entwickelt werden.

# 6 Kompletter elementweiser Ansatz

Mit Hilfe der Analysen aus Kapitel 4 und 5 wurde eine umfangreiche Grundlage geschaffen, die es erlaubt, komplette visuelle Ansätze zu entwickeln. Dabei orientiert sich die Entwicklung wie im vorherigen Kapitel an den Anforderungen des Systems und dessen Gestaltung und wird diese Schritt für Schritt abarbeiten, um letztlich zu einem Gesamtergebnis zu kommen. In diesem sechsten Kapitel wird der erste visuelle Ansatz entwickelt, der sich auf eine elementweise Darstellung und Navigation von Produktdaten fokussiert.

## 6.1 Darstellung der einzelnen abgefragten Objekte aus der CIO

Wie schon zu Beginn des letzten Kapitels muss als Erstes die Darstellung eines einzelnen abgefragten Objekts aus der CIO festgelegt werden. Um sich nur auf das Nötigste zu beschränken, werden die Objekte mit ihrem Namen zentriert in einer rechteckigen Umrandung angezeigt (vgl. Abbildung 6.1).


Abbildung 6.1: Einzelne Objekte werden mit ihrem Namen in einem Rechteck mit leichtem Grauton und schwarzem Rand dargestellt.

Damit kommt man dem gesteckten Ziel, nämlich einer schlichten, einfachen und übersichtlichen Darstellung der Elemente am nächsten. Um sich etwas vom weißen Systemhintergrund

abzuheben, besitzt die Fläche jedes Objekts einen leichten Grauton und einen schwarzen Rand.

## 6.2 Ansicht der Artefaktattribute

Nun muss entschieden werden, ob die Artefaktattribute standardmäßig oder erst auf Wunsch des Nutzers angezeigt werden. Für diesen endgültigen Ansatz wurde eine Darstellung auf Wunsch gewählt, da eine nähere Beschreibung der Objekte durch deren Attribute lediglich Zusatzinformationen sind. Um die spätere Gesamtdarstellung weder verschieben noch mit dem gesamten Objekt überdecken zu müssen, wurde ein eigenes kleines Fenster als elementweiser Darstellungsrahmen gewählt. Geöffnet wird dieses Fenster mit Hilfe einer Schaltfläche innerhalb eines Funktionsmenüs, welches durch einen Mausklick auf ein Element nahe daran erscheint (vgl. Abbildung 6.2).

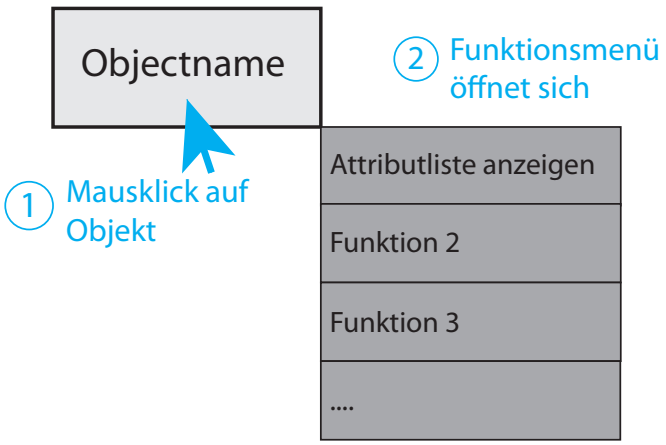


Abbildung 6.2: Durch Klick auf ein Objekt öffnet sich das elementweise Funktionsmenü.

Nach Klick auf die entsprechende Schaltfläche (vgl. Abbildung 6.3) werden die Attribute als einspaltige Liste innerhalb eines neuen Fensters dargestellt (vgl. Abbildung 6.4). Sollte der Platz für den Text eines Attributs nicht ausreichen, wird beim Klick auf das entsprechende Attribut ein Tooltip mit dem gesamten Text angezeigt. Außerdem bietet eine Scrollbar am rechten Rand die Möglichkeit, die Attributliste zu scrollen. Dies verhindert ein zu großes Fenster bei Objekten mit vielen Attributen.

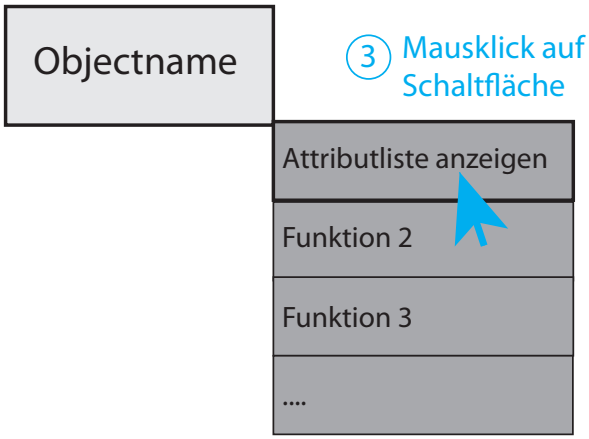


Abbildung 6.3: Das Funktionsmenü bietet die Möglichkeit, die Attributliste des Elements anzuzeigen.

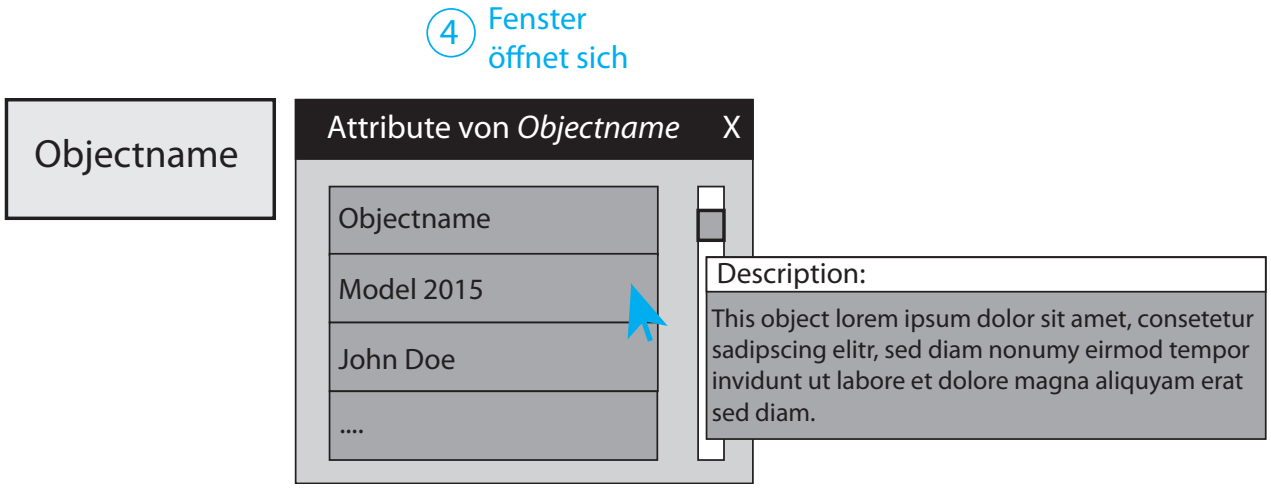


Abbildung 6.4: Durch Klick auf die Schaltfläche in Abbildung 6.3 öffnet sich ein neues kleines Fenster, in dem die Attributliste des Elements dargestellt ist. Bei längeren Einträgen hilft ein Tooltip, das den kompletten Textinhalt anzeigt.

## 6.3 Geeignete Gesamtdarstellung aller Ergebniselemente

Zur Erhaltung der Systemkonsistenz und um den Darstellungsplatz optimal auszunutzen, werden die Elemente als Gesamtes ebenfalls als ein großes Rechteck dargestellt (vgl. Abbildung 6.5).

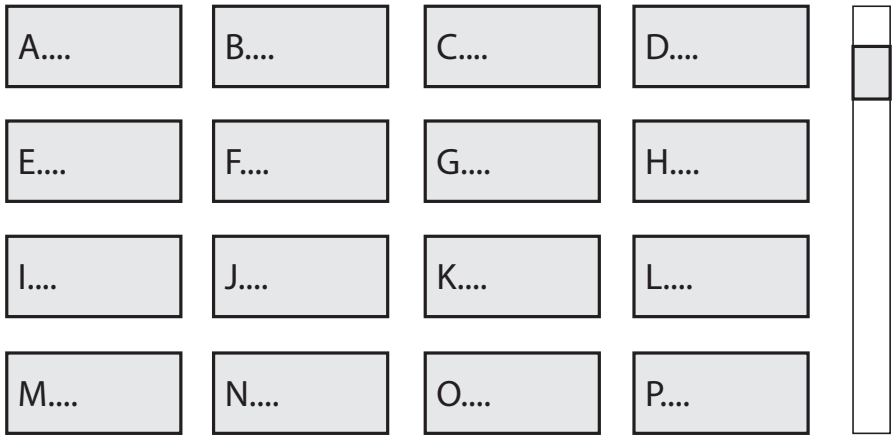


Abbildung 6.5: Für diesen elementweisen Ansatz wurde eine geordnete, rechteckige Gesamtdarstellung gewählt. Dabei werden die einzelnen Elemente alphabetisch von links nach rechts auf der Fläche platziert.

Die Flächeninhaltsanalyse in Abschnitt 5.3.2 ergab, dass diese geometrisch geordnete Form für diesen Kontext am besten geeignet ist. Ist das Ergebnis einer Datenabfrage klein, so wird das eigentliche Rechteck von links nach rechts mit den Elementen aufgefüllt. Als Kriterium für die Reihenfolge der Elemente dient das Alphabet beziehungsweise der erste Buchstabe des Ergebniselements. Dabei werden aufeinanderfolgende Elemente nebeneinander platziert. Da ein Ergebnis viele Elemente beinhalten kann, können diese mit Hilfe einer Scrollbar am rechten Rand innerhalb ihres Rahmens (siehe Abschnitt 6.4) durchgescrollt werden. Hierbei wurde sich gegen die Idee einer zweifachen Scrollbar (horizontal und vertikal) entschieden, um dem Benutzer innerhalb der Anzeigenfläche nicht unnötig die Orientierung zu erschweren.

## 6.4 Geeigneter Rahmen für eine Gesamtdarstellung

Als geeigneter Rahmen für die Darstellung einer Ergebnismenge wurde für den ersten visuellen Ansatz ein Fenster als optimale Lösung ausgemacht (vgl. Abbildung 6.6). Ein Fenster

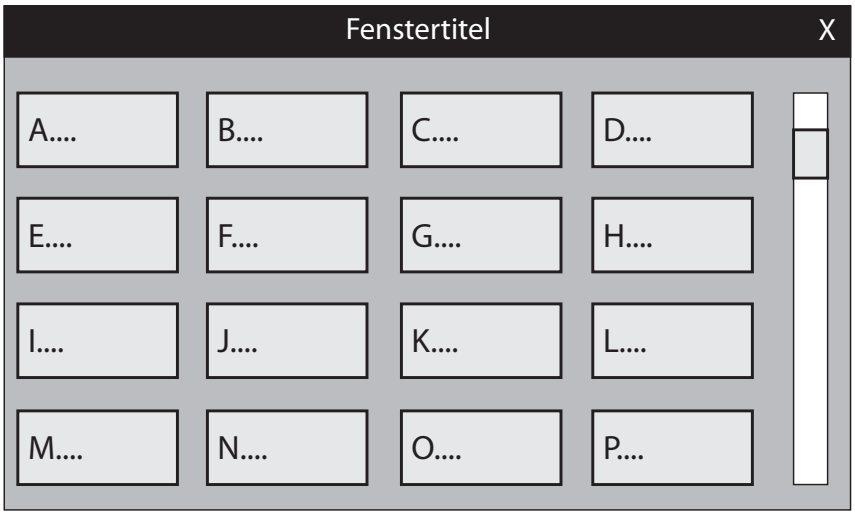


Abbildung 6.6: Als Rahmen dient in diesem Ansatz ein Fenster, in dem die Gesamtdarstellung der Ergebniselemente angezeigt wird.

bietet im Vergleich zu den anderen in Kapitel 5 genannten Alternativen vor allem die besten Vergleichsmöglichkeiten, da man mehrere davon gleichzeitig am Bildschirm platzieren und betrachten kann. Diese lassen sich mehrfach nebeneinander beziehungsweise übereinander am Bildschirm platzieren und gleichzeitig betrachten. Dadurch wird bestmöglich auf die Aufgabe des *Business Users*, die hauptsächlich die Analyse des Ergebnisses sein wird, Rücksicht genommen. Bezüglich des Designs wird zur besseren Unterscheidbarkeit der Rahmen des Fensters in schwarz gehalten und dessen Titel in weißer Schrift zentriert platziert. Die Größe des Fensters kann beliebig und individuell vom Benutzer durch Interaktion am Fensterrand verändert werden.

## 6.5 Darstellung der korrespondierenden Elemente aus Local Ontologies

Neben der eigentlichen Darstellung der Ergebnismenge gibt es noch weitaus mehr Anforderungen, die das System erfüllen muss. Dazu gehören unter anderem auch die korrespondierenden Elemente, die zu jedem Ergebniselement gehören.

### 6.5.1 Darstellungsort

Der Darstellungsort der korrespondierenden Elemente ist passend zur Attributliste (vgl. Abschnitt 6.2) ein eigenes Fenster (vgl. Abbildung 6.7). Dabei gelten die gleichen Bestimmungen wie bei dem Fenster für die Ergebniselemente. Das Fenster öffnet sich nach einem Mausklick auf das Objekt und der darauf folgenden Auswahl einer Schaltfläche des erscheinenden Funktionsmenüs. Wie genau die korrespondierenden Elemente innerhalb des Fensters dargestellt werden, wird erst in einem späteren Abschnitt definiert.

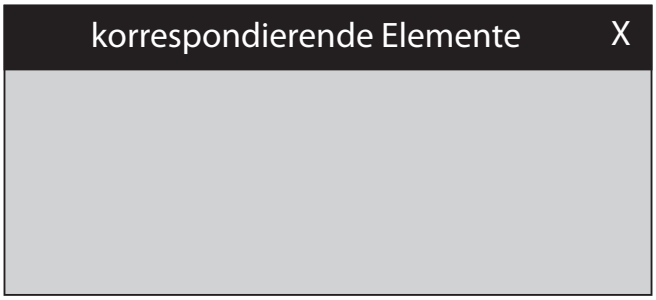


Abbildung 6.7: Auch für die korrespondierenden Elemente wurde ein eigenes Fenster als Darstellungsort gewählt.

### 6.5.2 Dargestellte Informationen

Was die Informationen des jeweiligen korrespondierenden Elements betrifft, so unterscheiden sich die Angaben kaum von denen der Ergebniselemente. Korrespondierende Objekte werden ebenfalls sowohl durch ihren Namen als auch durch einen rechteckigen Rahmen

repräsentiert. Zusätzlich wird jedoch bei den korrespondierenden Elementen die jeweilige Zugehörigkeit zur Local Ontology direkt unter dem Objektnamen angezeigt (vgl. Abbildung 6.8). Der Grund für diese Entscheidung ist die bessere und schnellere Zuordnung der einzelnen korrespondierenden Elemente zu ihrer „Umgebung“ durch den Benutzer. Dieser erkundigt sich nämlich nur dann nach korrespondierenden Elementen zu einem Ergebniselement, wenn er nähere Informationen zu dessen dazugehörigen Produktdaten analysieren will. Diese Aufgabe wird durch die zusätzliche Anzeige der Local Ontology wesentlich erleichtert.


Abbildung 6.8: Von den korrespondierenden Elementen wird nicht nur der Elementname angezeigt, sondern auch die Angabe, aus welcher Local Ontology es stammt.

Andernfalls müsste der Benutzer erst einen weiteren Dialog oder Ähnliches öffnen, um an diese kleine, aber doch recht wichtige Information zu gelangen. Um ihm diese Zeit zu sparen, macht es Sinn, ihm diese Information direkt mit anzuzeigen. Da der Name des Elements jedoch wichtiger ist, wird dieser in einer größeren Schrift angezeigt. Möchte man auch bei den korrespondierenden Elementen auf die Attribute zugreifen, so öffnet sich auch hier nach einem Mausklick auf ein Element das elementweise Funktionsmenü mit der Schaltfläche „Attributliste anzeigen“ (vgl. Abbildung 6.9). Dadurch öffnet sich wie schon in Abschnitt 6.2 ein eigenes kleines Fenster, in dem die Attribute ablesbar sind.

### 6.5.3 Gesamtdarstellung

Um erneut das Designkriterium der Konsistenz zu erfüllen, wird sich auch die Gesamtdarstellung der korrespondierenden Elemente an einer rechteckigen Form orientieren. Dabei gelten die gleichen Bestimmungen zur Reihenfolge und Position, welche schon in Abschnitt 6.3 definiert wurden.

### 6.5.4 Zeitpunkt der Darstellung

Da es sich bei den korrespondierenden Elementen um eine Zusatzinformation der eigentlichen Ergebnisdarstellung handelt, wird diese wie die Attributliste auch erst auf Wunsch des Benutzers angezeigt. Durch diesen geeigneten Umstand wird das Funktionsmenü aus Abschnitt 6.2 einfach um eine weitere Funktion erweitert (vgl. Abbildung 6.9).

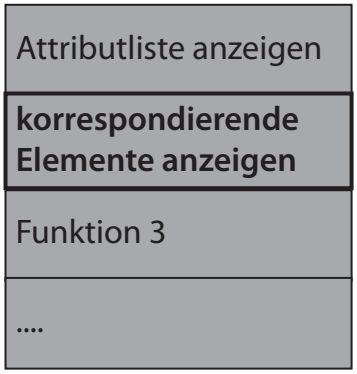


Abbildung 6.9: Das elementweise Funktionsmenü wird um die Funktion erweitert, um die korrespondierenden Elemente anzeigen zu können.

### 6.5.5 Darstellungsart

Durch die Entscheidung für eine weitere Schaltfläche im elementweisen Funktionsmenü ist auch die Frage der globalen oder elementweisen Darstellung der korrespondierenden Elemente für diesen Ansatz geklärt. Um den *Business User* nicht mit zu vielen Informationen zu überfluten und aufgrund der Annahme, dass er sich grundsätzlich leichter tut, wenn er nur für ein Element gleichzeitig nähere Informationen bekommt, wurde sich für diese Art der Darstellung entschieden. Die Abbildungen 6.10, 6.11 und 6.12 zeigen den Ablauf der Interaktion, mit der sich der Benutzer die korrespondierenden Elemente für ein Objekt anzeigen lassen kann. Der Inhalt des Fensters, wie es Abbildung 6.12 zeigt, ist nur vorerst so dargestellt. Genaueres dazu wird erst in Abschnitt 6.6 definiert.

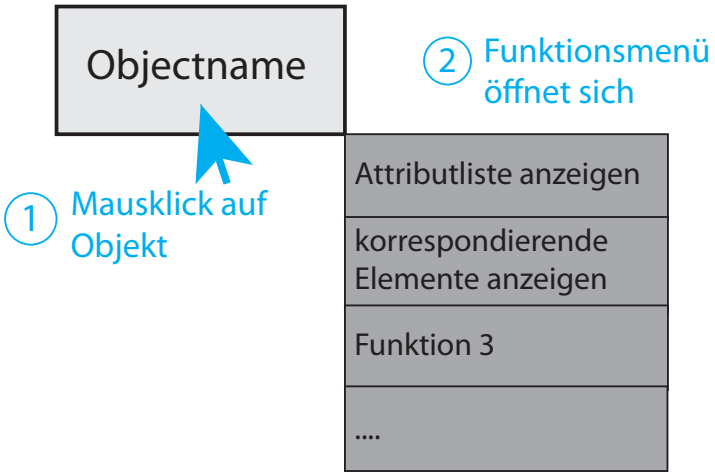


Abbildung 6.10: Durch Klick auf ein Element wird dessen Funktionsmenü angezeigt.

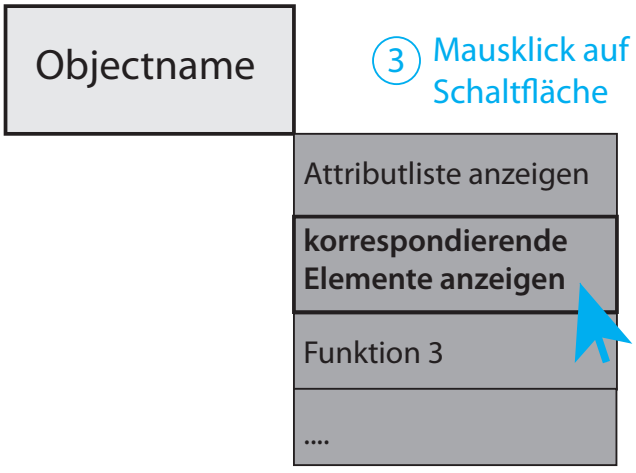


Abbildung 6.11: Die neue Funktion im Funktionsmenü ermöglicht die Anzeige derjenigen korrespondierenden Elemente, die zu dem angeklickten Element gehören.

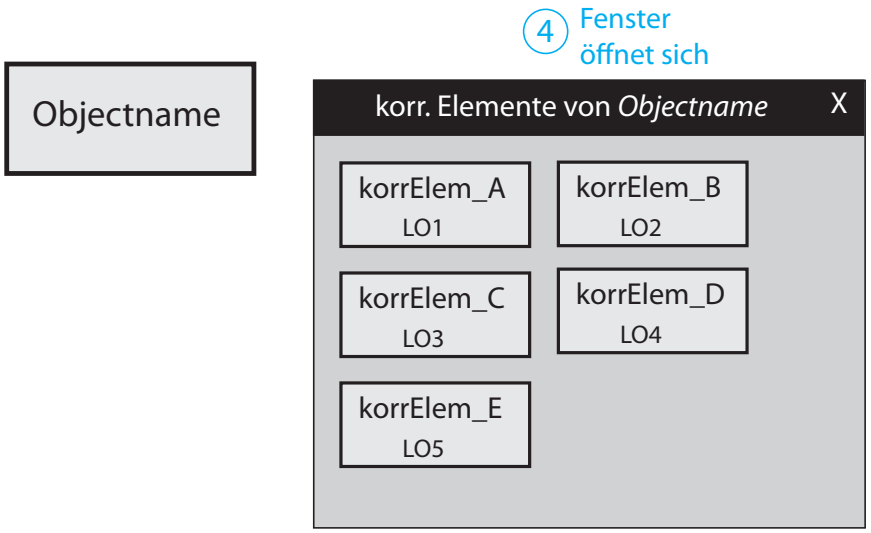


Abbildung 6.12: Das nach dem angeklickten Element benannte Fenster für die Anzeige der korrespondierenden Elemente öffnet sich nach Klick auf die Schaltfläche aus Abbildung 6.11.

### 6.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen

Aufgrund der Wahl einer elementweisen Darstellung der korrespondierenden Elemente ist eine extra Darstellung der Zusammengehörigkeit nicht notwendig.

### 6.5.7 Berücksichtigung von mehrwertigen Korrespondenzen

In diesem endgültigen Ansatz spielt auch der Punkt der mehrwertigen Korrespondenzen keine Rolle. Besteht eine 1:n Beziehung zwischen Ergebniselement und korrespondierenden Elementen, so werden alle *n* korrespondierenden Artefakte im entsprechenden Fenster angezeigt (vgl. Abbildung 6.13). Bei einer n:1 Beziehung tritt das eine korrespondierende Element sowohl im passenden Fenster für Ergebniselement A als auch B auf (vgl. Abbildung 6.14).

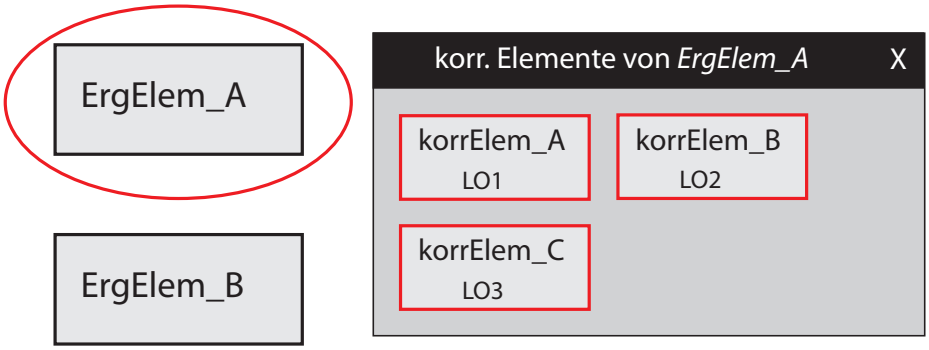


Abbildung 6.13: 1:n Beziehung zwischen Ergebniselement und korrespondierenden Elementen.

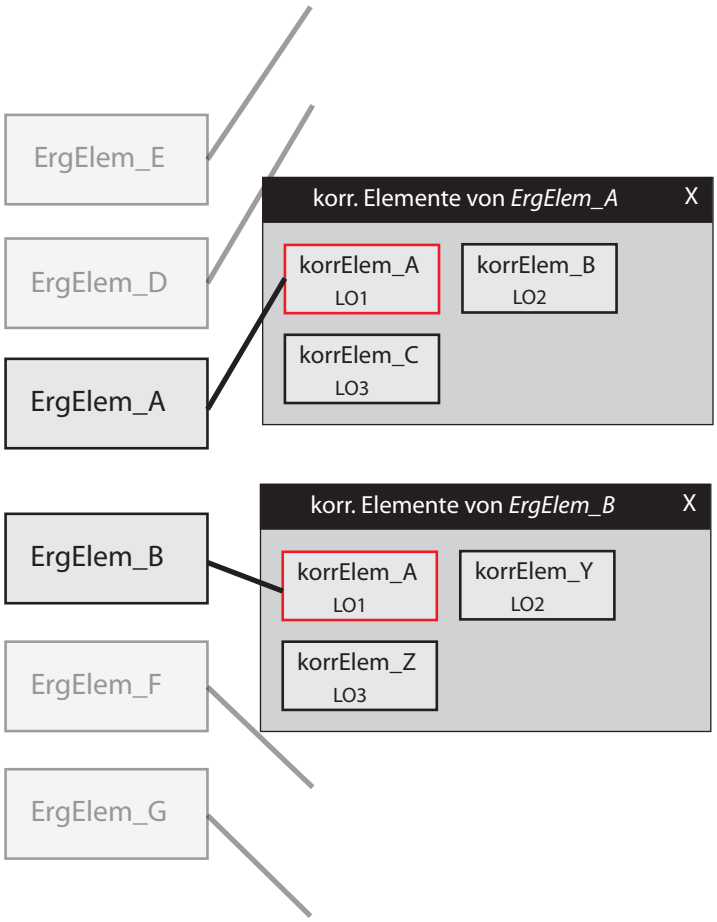


Abbildung 6.14: n:1 Beziehung zwischen Ergebniselement und korrespondierenden Elementen.

## 6.6 Ebenenwechsel

Während sich alle bisherigen Schritte einzig und allein auf die Objektebene bezogen, kommt in diesem Abschnitt noch hinzu, dass es nicht nur Objekte, sondern dazu auch verschiedene Varianten und Versionen gibt. Wie man diese passend zu diesem Ansatz berücksichtigen und einen Ebenenwechsel ermöglichen kann, zeigen die folgenden Unterkapitel.

### 6.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten

Bei der Darstellung der einzelnen Varianten und Versionen orientiert sich dieser Ansatz an dem Beispiel aus Abschnitt 5.6.1. Dieser definiert für Varianten und Versionen die Verwendung des Elementnamens in einer Kreis- beziehungsweise Sechseckform (vgl. Abbildung 5.20). Da sich die Elemente lediglich in ihrer Ebene unterscheiden, müssen für sie keine Erweiterungen oder Einschränkungen bestimmt werden. Sie verhalten sich wie die oben beschriebenen Objekte und können auch wie diese angeklickt werden, wodurch sich das elementweise Funktionsmenü aus Abbildung 6.9 öffnet. Mit Hilfe dieses Funktionsmenüs kann, wie bei den Objekten auch, der Zugriff auf die Attributliste des jeweiligen Elements erfolgen.

### 6.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen

Im Hinblick auf die Konsistenz und die Flächeninhaltsanalyse aus Abschnitt 5.3.2 wird auch in den tieferen Ebenen auf ein Rechteck als Gesamtdarstellungsform zurückgegriffen. Bei den gewählten Formen für Varianten und Versionen wird innerhalb dieser Gesamtdarstellung noch verhältnismäßig wenig Darstellungszwischenraum (schwarz) vergeudet (vgl. Abbildung 6.15). Bei einer Dreiecksdarstellung beispielsweise wird mindestens die Hälfte der Fläche, in die wiederum Dreiecke passen würden, zu Ungunsten der Gesamtdarstellung verschwendet.

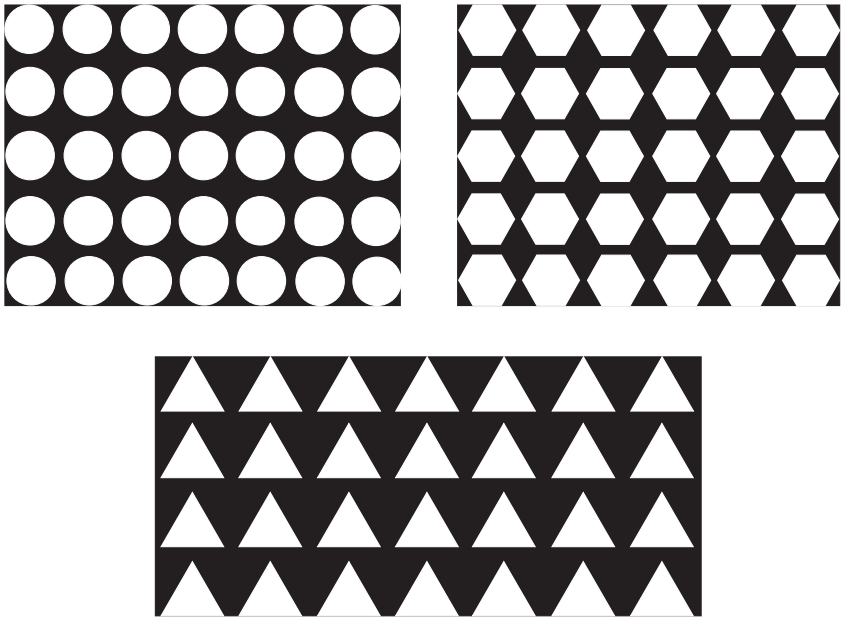

Abbildung 6.15: Darstellung des verschwendeten Darstellungsplatzes (schwarz) je nach Wahl der Darstellungsform der einzelnen Elemente (weiß).

### 6.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen

Varianten und Versionen sind grundlegende, wichtige Informationen. Trotzdem werden auch diese in ein eigenes Fenster ausgelagert, um ausreichend Darstellungsplatz für die Vielzahl an Informationen zu ermöglichen (vgl. Abbildung 6.16).

### 6.6.4 Art des Ebenenwechsels

Ähnlich zu den korrespondierenden Elementen wird auch der Ebenenwechsel in diesem Ansatz elementweise gelöst. Dadurch wird einfach eine weitere Schaltfläche „andere Ebenen anzeigen“ im Funktionsmenü eingefügt (vgl. Abbildung 6.17). Durch Klick auf diese Schaltfläche öffnet sich das Fenster aus Abbildung 6.16 mit einem Tab für jeweils beide anderen Ebenen des angeklickten Elements (hier: Objektebene, vgl. Abbildung 6.18).

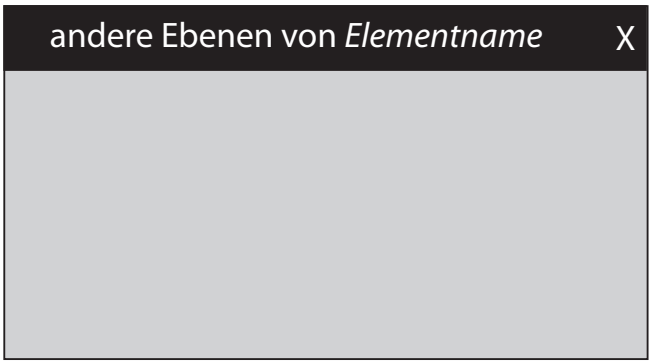


Abbildung 6.16: Um die Konsistenz des Frameworks zu bewahren, wird auch für die anderen Ebenen ein Fenster als Darstellungsrahmen gewählt.

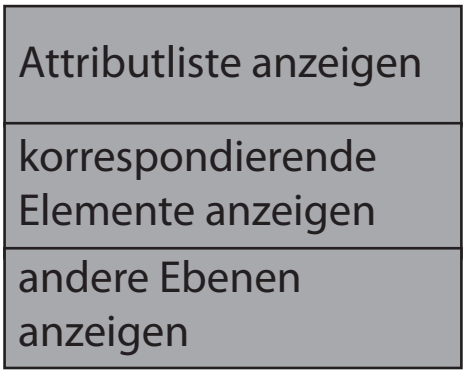


Abbildung 6.17: Das endgültige Funktionsmenü für diesen elementweisen Ansatz beinhaltet als dritte Funktion die Möglichkeit, die anderen Ebenen eines angeklickten Elements anzuzeigen.

### 6.6.5 Ebenen der korrespondierenden Elemente

Da korrespondierende Elemente letztlich nichts anderes als die Elemente eines Ergebnisses sind, kann der im vorherigen Abschnitt definierte elementweise Ebenenwechsel dafür ebenfalls angewendet werden. Denn letztlich möchte der Benutzer nichts anderes, als sich tiefergehend für dieses spezielle Element zu informieren. Ob es sich dabei um ein abgefragtes oder ein korrespondierendes Element handelt, ist dabei irrelevant. Bezüglich der Form der Elemente wird ebenfalls auf die Unterscheidung verwiesen, die schon für die Ergebniselemente definiert wurde, damit insgesamt ein einheitlicher und konsistenter Eindruck des Systems erweckt wird.

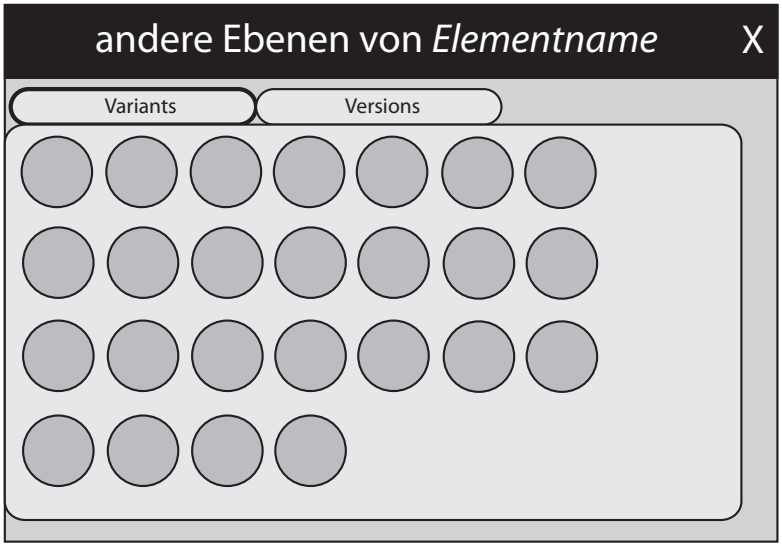


Abbildung 6.18: Zur Darstellung der Elemente der beiden anderen Datenebenen wurde für diesen Ansatz ein Fenster mit einem Tab für jeweils beide Ebenen.

Nach Abarbeitung aller Schritte dieses Kapitels existiert nun ein kompletter visueller Ansatz, der die Anforderungen der Benutzer- und Aufgabenanalyse berücksichtigt und auf eine elementweise Arbeitsweise ausgerichtet ist. Außerdem wurden gestaltungstechnische Kriterien wie Übersichtlichkeit, Komfort und Konsistenz in diesen Ansatz integriert. All das sollte dem Benutzer eine bestmögliche Darstellung bieten, damit er seine Aufgaben gerecht bewerkstelligen kann. Im nächsten Kapitel wird ein Ansatz beschrieben, der statt einer elementweisen auf eine globale Herangehensweise abzielt.

# 7 Kompletter globaler Ansatz

Der folgende Ansatz dieses Kapitels ist vor allem danach ausgerichtet, dass der Benutzer einen schnell zugreifbaren, kompletten Überblick über das Ergebnis seiner Abfrage bekommt. Dabei sollte es ihm weniger um die tiefergehenden Informationen einzelner Elemente gehen, sondern eher darum, welche Elemente im Hinblick auf die entsprechende Abfrage betroffen sind.

## 7.1 Darstellung der einzelnen abgefragten Objekte aus der CIO

Da sich die Darstellung der Objekte als Rechteck mit dem Objektname als zentrierte Information als gut erwiesen hat, wird sie auch für diesen Ansatz gewählt.

## 7.2 Ansicht der Artefaktattribute

Anders als beim ersten Ansatz wird in diesem nicht nur der Name eines Elements dargestellt, sondern zusätzlich auch ein Teil der dazugehörigen Attribute (vgl. Abbildung 7.1). Mit Hilfe einer Scrollbar kann durch die Liste gescrollt werden, um die restlichen Attribute sichtbar werden zu lassen. Zusätzlich gibt es auch hier wieder die Möglichkeit, einzelne Attribute anzuklicken, um die vollständige Beschreibung des jeweiligen Attributs in einem Tooltip angezeigt zu bekommen.

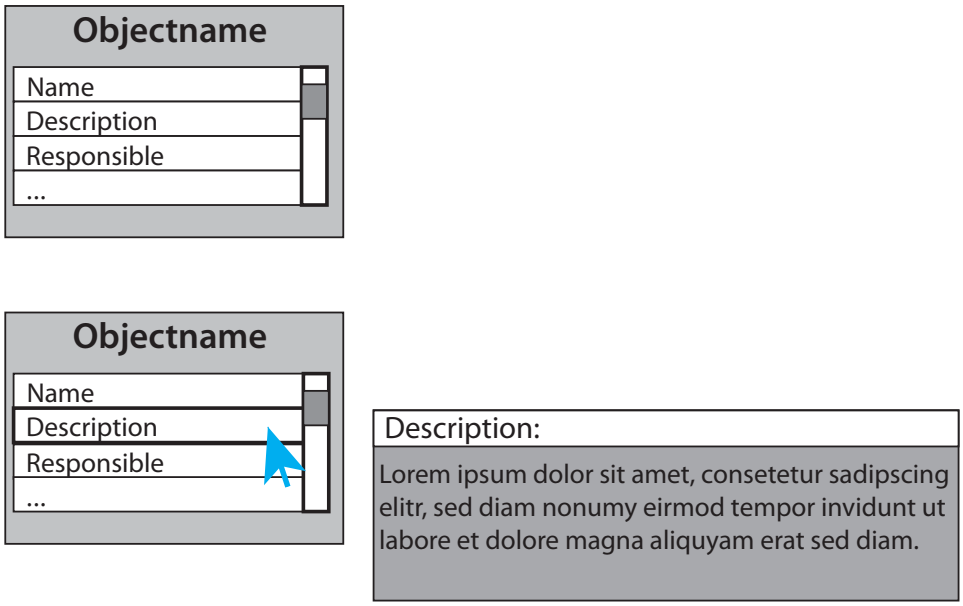


Abbildung 7.1: Für diesen Ansatz wurde erneut eine rechteckige Darstellung der einzelnen Objekte gewählt. Zusätzlich dazu wird gleichzeitig die Attributliste mit anklickbaren Attributeinträgen angezeigt.

## 7.3 Geeignete Gesamtdarstellung aller Ergebniselemente

Als geeignete Gesamtdarstellung wird auch in diesem Ansatz ein Rechteck verwendet. Alle weiteren Bestimmungen und Einschränkungen entsprechen den Festlegungen aus Abschnitt 6.3

## 7.4 Geeigneter Rahmen für eine Gesamtdarstellung

Für den Rahmen der Gesamtdarstellung bietet dieser Ansatz eine kleine Besonderheit, denn es wird weder auf Fenster, noch auf Tabs zurückgegriffen. Der Rahmen dieses Ansatzes befindet sich innerhalb des Systems in einer eigens dafür vorgesehenen Anzeige. Diese interne Anzeige bietet am oberen Ende eine Schaltfläche für bestimmte Befehle, die im weiteren Verlauf näher betrachtet werden. Als zentraler Punkt dieser Anzeige wird anfangs über den kompletten Platz des Rechteckes die gesamte Ergebnismenge platziert (vgl. Abbildung

7.2). Als Navigationshilfsmittel zur Übersicht aller Elemente wird in diesem konkreten Fall auf eine Seitenansicht zurückgegriffen. Zusätzlich gibt es ebenfalls die Möglichkeit, durch die Schaltfläche „Attribute verbergen“ im globalen Menü, die Attributlisten der Elemente zu verbergen, falls der Benutzer eine aufgeräumtere, informationsärmere Ansicht wünscht. Durch einen Mausklick auf diese Schaltfläche verschwinden die Attributlisten und es bleibt lediglich der Name der Elemente sichtbar.

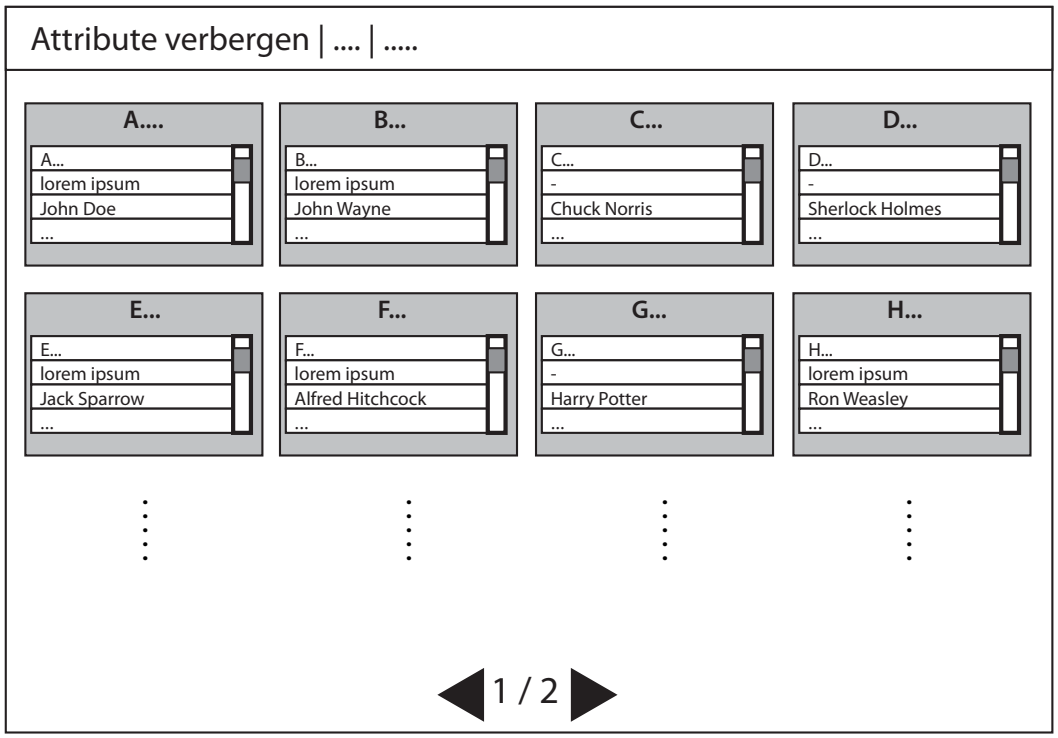


Abbildung 7.2: Der Rahmen dieses Ansatzes befindet sich innerhalb des Systems in einer eigens dafür vorgesehenen Anzeige. Diese interne Anzeige bietet am oberen Ende eine Schaltfläche für bestimmte Befehle. Zusätzlich wird über eine Seitenansicht eine Navigation durch alle Elemente ermöglicht.

## 7.5 Darstellung der korrespondierenden Elemente aus Local Ontologies

Bis hierhin wurde festgelegt, wie die einzelnen Ergebniselemente aussehen, welche Informationen diese beinhalten und wie sie als Gesamtes dargestellt werden. Außerdem wurden ihr Darstellungsort und der Zugriff darauf über mehrere Seiten definiert. In diesem Abschnitt wird nun festgelegt, in welcher Art und Weise die korrespondierenden Elemente aus Local Ontologies angezeigt werden.

### 7.5.1 Darstellungsort

Der Darstellungsort der korrespondierenden Elemente muss in diesem konkreten Fall erst konstruiert werden. Dies bedeutet, dass platztechnisch anfangs das Augenmerk des Benutzers auf die Ergebnismenge gerichtet sein soll. Möchte er die korrespondierenden Elemente abrufen, so muss zuerst die Ansicht verändert werden. Während die bisherige Anzeige der Ergebniselemente wie in Abbildung 7.2 über den gesamten Bildschirm ging, bleibt dafür am Ende nur noch die linke Hälfte des Bildschirms übrig, d.h. die Anzeige wird zusammengestaucht (vgl. Abbildung 7.3).

Der rot markierte Bereich auf der rechten Seite in Abbildung 7.3 stellt dabei den Darstellungsraum für die korrespondierenden Elemente dar. Bei dieser Umsetzung ist zu beachten, dass sich die Seitenanzahl der Ergebniselemente erhöhen kann, da weniger Platz pro Seite für deren Darstellung vorhanden ist. Außerdem verschieben sich die Elemente, weshalb sich der Benutzer kurz neu orientieren muss, falls er sich auf ein bestimmtes Element konzentriert hat. Aufgrund der alphabetischen Reihenfolge sollte dies jedoch kein wirkliches Problem sein. Dieser kleine Nachteil kann auch ohne größere Bedenken eingegangen werden, denn die Möglichkeit, mit diesem Darstellungsort beide Hauptbestandteile eines Abfrageergebnisses direkt nebeneinander zu haben, ist ein erheblicher Vorteil gegenüber anderen erwähnten Möglichkeiten. Außerdem wäre denkbar, den mittig platzierten Trennbalken als eine Art Schieberegler zu realisieren, den der Benutzer sowohl nach links als auch nach rechts schieben kann, um eine der beiden Ansichten komplett am Bildschirm zu haben. Standardmäßig gilt die komplette Breite nämlich lediglich für die Ergebnismengen-Ansicht.

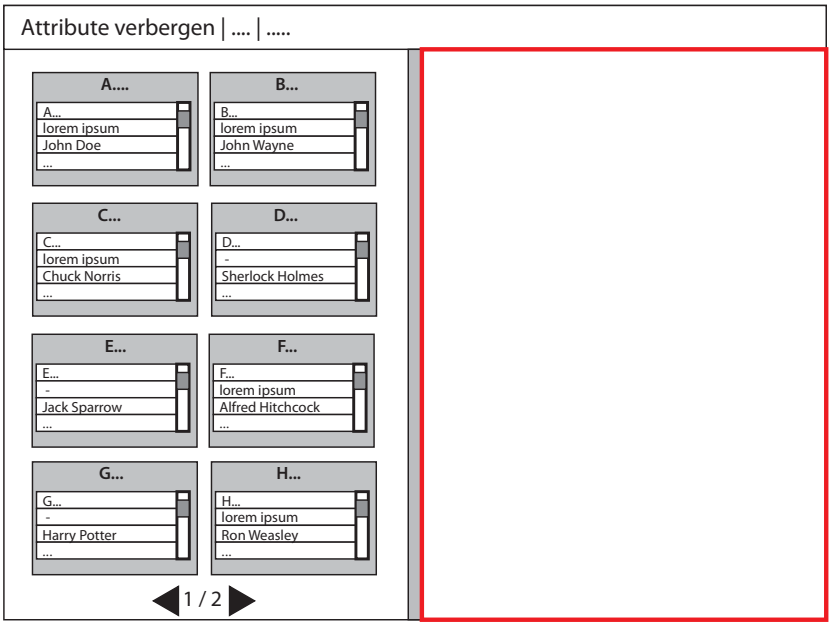


Abbildung 7.3: In diesem globalen Ansatz ist links der Platz für die Ergebniselemente und rechts der Platz für die korrespondierenden Elemente. Dieser Platz entsteht jedoch erst nach Interaktion des Benutzers.

### 7.5.2 Dargestellte Informationen

Da dem Benutzer mit diesem Ansatz gleich einige Informationen angeboten werden sollen, bietet es sich an, die korrespondierenden Elemente ebenso wie die Objekte mit Objektnamen samt Attributliste darzustellen (vgl. Abbildung 7.4). Dabei wird die erste Zeile jeder Attributliste die Angabe sein, zu welcher Local Ontology das entsprechende Element gehört. Damit unterscheiden sich Ergebnisobjekte und korrespondierende Objekte optisch nur von der ersten Attributzeile.

### 7.5.3 Gesamtdarstellung

Da die Ergebniselemente auf der linken Seite mehrspaltig (rechteckig) innerhalb einer beziehungsweise mehrerer Seiten angezeigt werden, macht es Sinn, diese Form auch für die Gesamtdarstellung der korrespondierenden Elemente zu wählen (vgl. Abbildung 7.5).

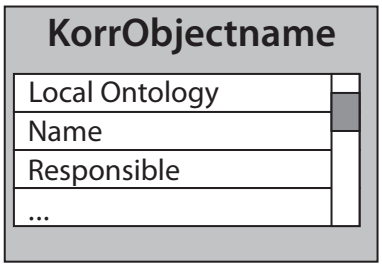


Abbildung 7.4: Auch die korrespondierenden Objekte werden rechteckig samt Attributliste dargestellt. Dabei gibt die erste Zeile an, aus welcher Local Ontology das Element stammt.

Dadurch entsteht ein konsistenter Gesamteindruck und der Benutzer kann beide Seiten ähnlich handhaben.

### 7.5.4 Zeitpunkt der Darstellung

Wie aus den oberen Abschnitten bereits erwähnt, werden die korrespondierenden Elemente nicht standardmäßig, sondern erst auf Wunsch des Benutzers angezeigt. Dies wird durch eine weitere Schaltfläche am oberen Rand des „Systemrahmens“ ermöglicht (vgl. Abbildung 7.6).

### 7.5.5 Darstellungsart

Da sich dieser Ansatz an einer globalen Ausrichtung orientiert, wird auch bei den korrespondierenden Elementen eine globale Darstellung gewählt. Diese wirkt sich insofern auf das System aus, als dass diese nicht mehr für jedes Element der Ergebnismenge, sondern nur noch für alle Elemente angezeigt werden. Dadurch erspart man dem Benutzer, vor allem bei einer großen Ergebnismenge, sich durch jedes Element zu klicken, um sich nach und nach alle betroffenen korrespondierenden Elemente zusammen zu suchen. Klickt der *Business User* folglich auf die Schaltfläche „korr. Elemente anzeigen“, so gleitet die Ergebnisanzeige nach links und rechts erscheinen alle betroffenen, korrespondierenden Elemente der Abfrage (vgl. Abbildung 7.7 - 7.9).

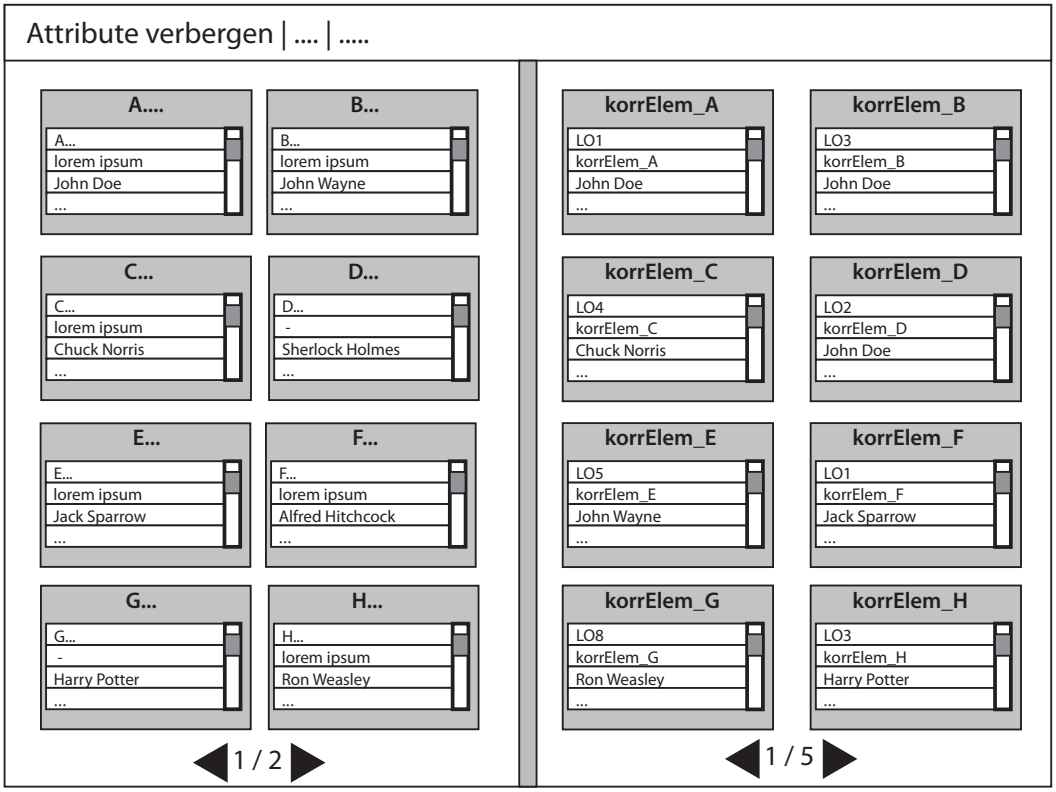


Abbildung 7.5: Um das Gesamtbild dieses globalen Ansatzes zu vervollständigen, werden die korrespondierenden Elemente in gleicher Art und Weise wie die Ergebniselemente platziert.

### 7.5.6 Zusammengehörigkeit von CIO- und korrespondierenden Elementen

Durch die Wahl einer globalen Anzeige der korrespondierenden Elemente kann eine Darstellung der Zusammengehörigkeit nur auf Kosten der Reihenfolge der korrespondierenden Elemente geschehen. Denn nur dadurch kann immerhin eine gewisse Gruppierung auf der rechten Seite geschaffen werden. Klickt man auf eines der Ergebniselemente, so wird dieses farbig markiert. Zusätzlich werden die dazugehörigen Elemente auf der rechten Seite in der gleichen Farbe markiert (vgl. Abbildung 7.10). Das System unterstützt den Benutzer insofern, als dass es an das erste korrespondierende Element der „Gruppe“ springt, die zu dem angeklickten Element auf der linken Seite gehört. Dabei kann sich der Benutzer so lange durch die Seiten klicken, bis kein Element mehr markiert ist. Dadurch wird angezeigt,

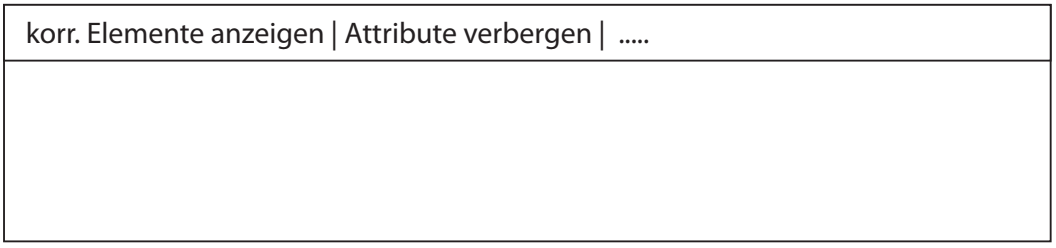


Abbildung 7.6: Die Schaltfläche „korr. Elemente anzeigen“ ermöglicht die Aufteilung der Bildschirmanzeige und der Platzierung der korrespondierenden Elemente.

dass er alle korrespondierenden Elemente seines angeklickten Artefakts gesehen hat. Klickt er wiederum ein korrespondierendes Element an, so wird dieses farblich markiert und die Anzeige auf der linken Seite springt auf das dazugehörige Ergebniselement, welches passend dazu markiert ist.

### 7.5.7 Berücksichtigung von mehrwertigen Korrespondenzen

Durch die farbliche Markierung spielt dieser Punkt wie auch schon beim ersten Ansatz keine Rolle.

## 7.6 Ebenenwechsel

Bis hierhin sind fast alle Anforderungen erfüllt, einzig und allein der Ebenenwechsel muss in diesen Ansatz noch integriert werden. Dieser wirkt sich sowohl auf die Ergebniselemente als auch auf die korrespondierenden Elemente aus. Wie auch schon in den vorherigen Abschnitten wird auch hier eine globale Herangehensweise bevorzugt.

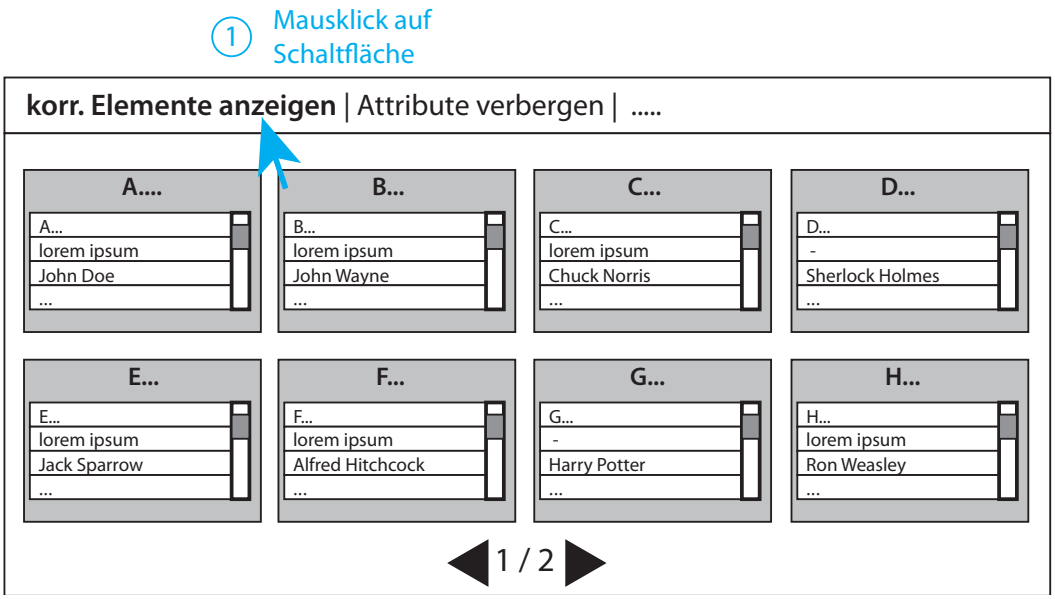


Abbildung 7.7: Um den Vorgang für die Anzeige der korrespondierenden Elemente zu starten, muss auf die Schaltfläche „korr. Elemente anzeigen“ geklickt werden.

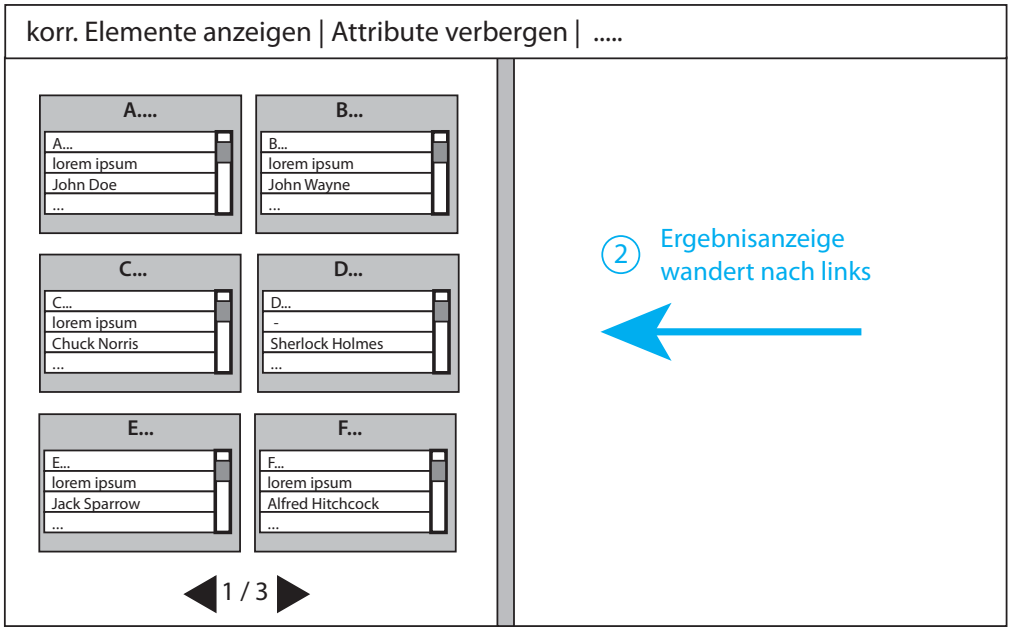


Abbildung 7.8: Durch den Klick auf die Schaltfläche wandert die Gesamtdarstellung der Ergebniselemente nach links und ein Balken trennt die Bildschirmanzeige mittig.

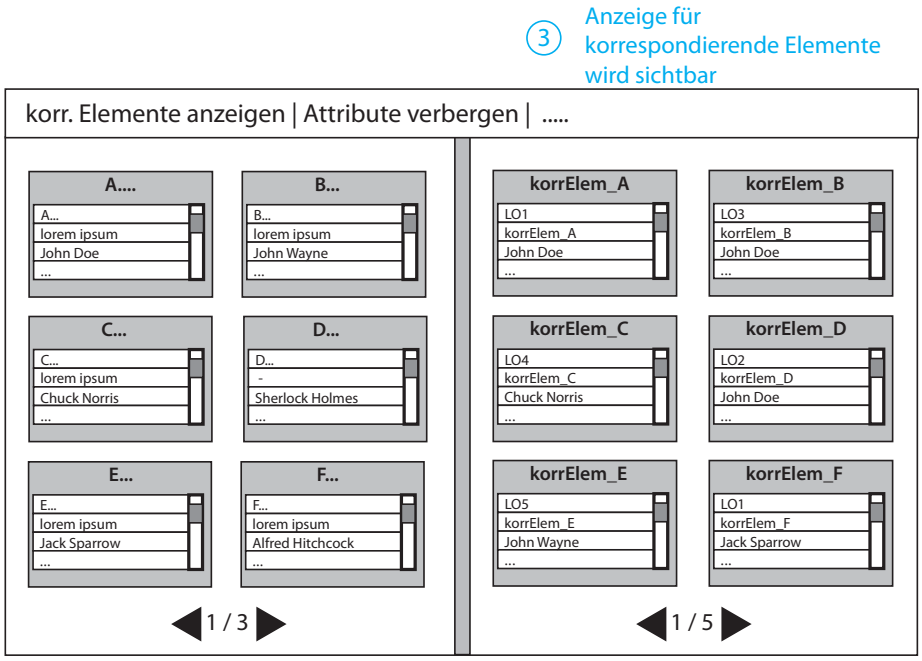


Abbildung 7.9: Als Ergebnis dieses Vorgangs erscheint auf der rechten Seite die Gesamtdarstellung der korrespondierenden Elemente.

### 7.6.1 Darstellung der einzelnen Varianten und Versionen samt Attributwerten

Obwohl sich die Kreis- beziehungsweise Sechseckform in den Analysen als nützlich erwiesen hat, wird darauf in diesem Ansatz aufgrund der zusätzlich angezeigten Attributliste verzichtet. Das bedeutet, dass sowohl die Objekte als auch die Varianten und Versionen der Ergebnismenge rechteckig mit Elementname sowie Attributliste angezeigt werden.

### 7.6.2 Geeignete Gesamtdarstellung der Varianten und Versionen

Bezüglich der Gesamtdarstellung wird ebenfalls auf die mehrspaltige beziehungsweise rechteckige Positionierung der Elemente wie schon bei den Objekten zurückgegriffen (vgl. Abbildung 7.2).

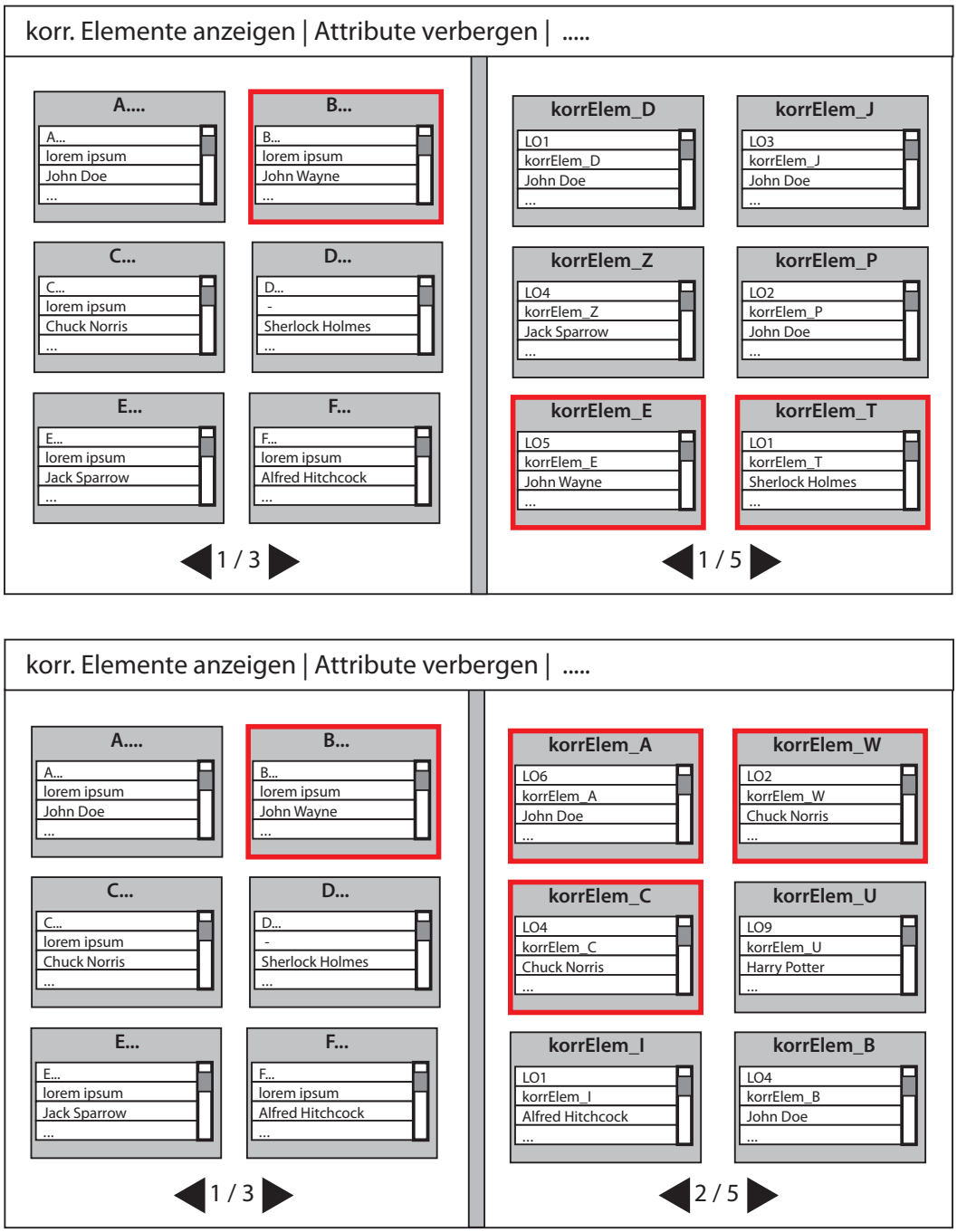


Abbildung 7.10: Durch farbliche Unterlegungen wurde in diesem globalen Ansatz die Zusammengehörigkeit zwischen Ergebniselementen und korrespondierenden Elementen geregelt.

### 7.6.3 Geeigneter Rahmen für Gesamtdarstellung der Varianten und Versionen

Für die Ergebnismenge wurde in diesem Ansatz anfangs die komplette Anzeige und bei gleichzeitiger Ansicht der korrespondierenden Elemente die linke Hälfte des Bildschirms gewählt. Da es sich bei Varianten und Versionen auch um Ergebniselemente handelt, ist ihr Darstellungsrahmen ebenfalls die gesamte beziehungsweise linke Hälfte der Bildschirmanzeige innerhalb der Seitenansicht. In diesen Seiten werden die Varianten und Versionen angezeigt. Dadurch kann sich der Benutzer gut merken, wo er die Ergebniselemente unabhängig von ihrer Ebene findet. Zur Unterscheidung der Ebenen dient eine Anzeige in einem Dropdown-Menü, auf welches im folgenden Abschnitt eingegangen wird.

### 7.6.4 Art des Ebenenwechsels

Wie schon in den vorigen Abschnitten mehrfach angemerkt, ist dieser Ansatz global ausgerichtet, sodass auch der Ebenenwechsel global vollzogen wird. Dies hat zur Folge, dass durch ein einfaches globales Dropdown-Menü im systeminternen Menü ausgewählt werden kann, welche Ebene sich der Benutzer anzeigen lassen möchte (vgl. Abbildung 7.11).

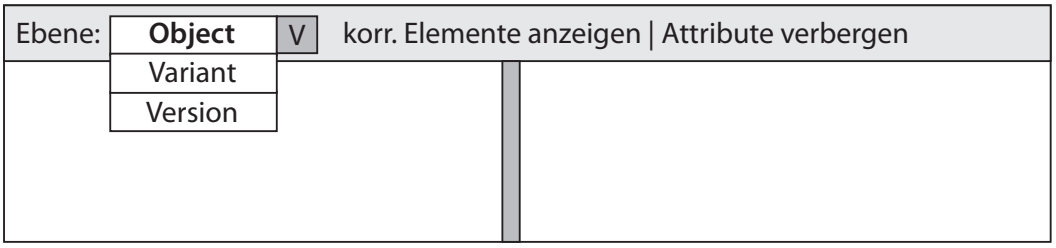


Abbildung 7.11: Durch ein einfaches globales Dropdown-Menü im systeminternen Menü kann zwischen den verschiedenen Ebenen gewählt werden.

Daraufhin werden alle Elemente der ausgesuchten Ebene im dafür vorgesehenen Rahmen dargestellt. In Abbildung 7.12 sind beispielsweise links alle Varianten der Datenabfrage angezeigt und gleichzeitig rechts die dazugehörigen korrespondierenden Elemente.

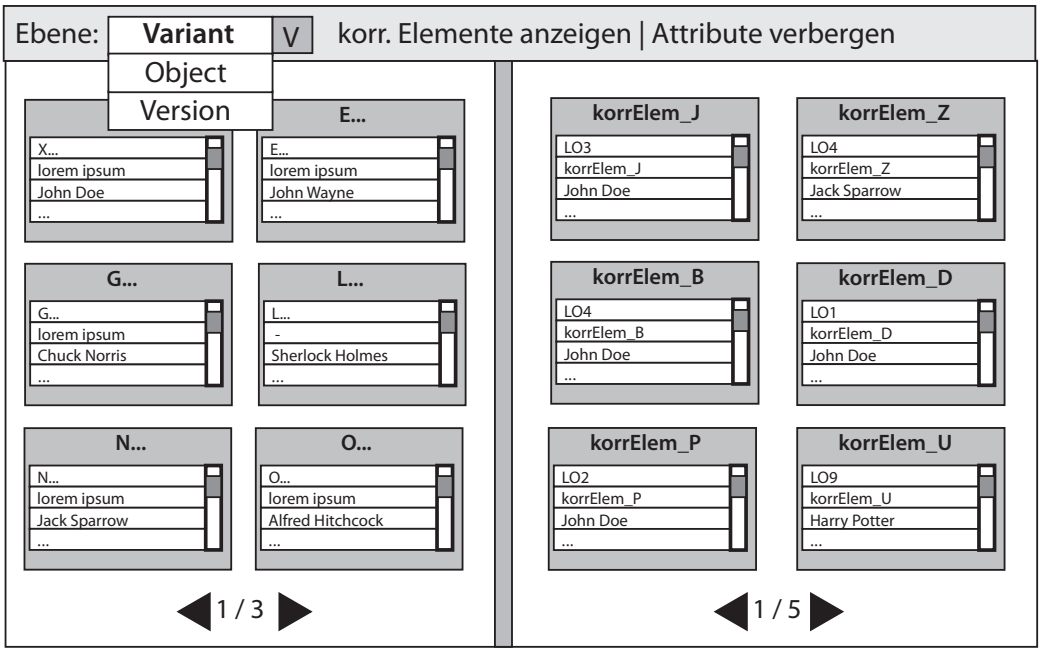


Abbildung 7.12: Beispielhafte Anzeige der Varianten von Ergebnis- und korrespondierenden Elementen.

### 7.6.5 Ebenen der korrespondierenden Elemente

Bezüglich der Form der unterschiedlichen Ebenen der korrespondierenden Elemente ist zu sagen, dass sich diese nicht voneinander unterscheiden. Aufgrund der Attributliste ist eine rechteckige Darstellung die optimalste, wodurch sich eine Änderung nur negativ auf das Design und die Darstellung auswirken würde. Somit ist der Benutzer angewiesen, sich an der Anzeige im Dropdown-Menü zu orientieren, um feststellen zu können, auf welcher Ebene er sich gerade befindet. Passend zur Ebene der Ergebnismenge wird das System bei Änderung der Ebene im Dropdown-Menü auch die Ebene der korrespondierenden Elemente ändern. Wechselt der Benutzer also beispielsweise von der Objektebene zur Variantenebene, so aktualisiert sich sowohl die linke als auch die rechte Ansicht.

Damit sind auch für diesen Ansatz alle notwendigen Anforderungen erfüllt und der Ansatz somit komplett. Statt einer elementweisen Darstellung und Navigation wurde hierbei mehr-

fach auf eine globale Ausrichtung hin entwickelt. Dadurch beinhaltet diese Arbeit nun zwei visuelle Ansätze für eine Darstellung und Navigation von komplexen Produktdaten aus zwei verschiedenen Blickwinkeln.

# 8 Realisierung und Evaluation

Mit den visuellen Ansätzen aus den Kapiteln 6 und 7 ist der wichtigste Teil dieser Arbeit für das PROCEED-Framework erfüllt. Mit diesem letzten Kapitel 8 wird die Phase *User-Interface-Entwurf* durch die Vorstellung von elektronischen UI-Prototypen abgeschlossen und es werden Aussagen über die Phase *Evaluationen und Tests* getroffen (vgl. Abbildung 8.1).

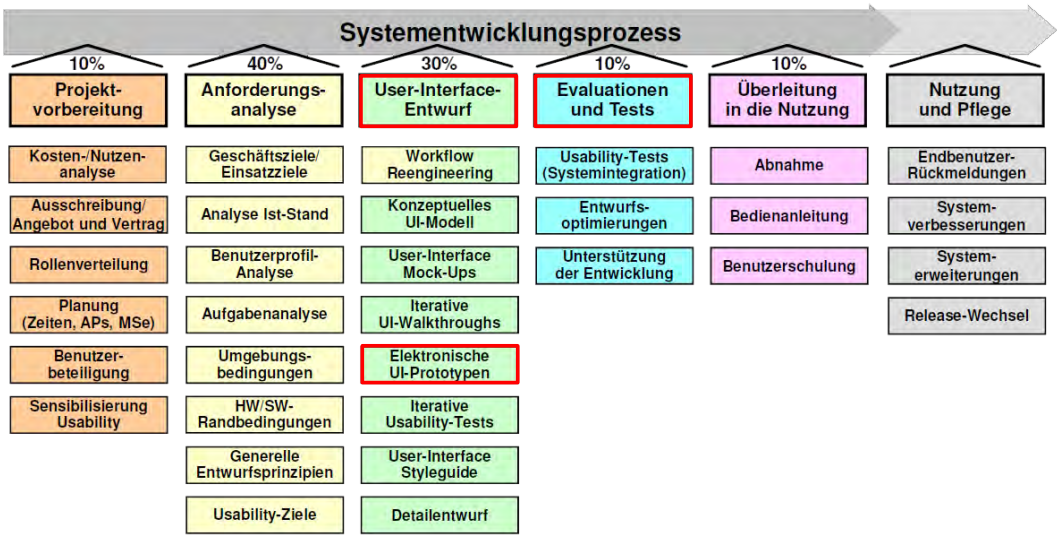


Abbildung 8.1: Für dieses Kapitel relevant ist der Prozessschritt „Elektronische Prototypen“ der Phase User-Interface-Entwurf und die Phase Evaluationen und Tests.

## 8.1 Elektronische UI-Prototypen

Sinn und Zweck des Prozessschritts *Elektronische UI-Prototypen* ist die Entwicklung von groben, interaktiven Benutzeroberflächen auf Basis der erstellten User-Interface-Mockups. Diese Protoypen sind lauffähig, beinhalten die wesentlichen Dialogelemente und lassen eine Navigation im System zu. Somit sind sie zwar bedienbar, jedoch fehlt ihnen eine detaillierte Anwendungsfunktionalität, die damit nur grob simuliert werden kann. Für diese Arbeit wurde dafür auf ein online Mockup, Wireframe und UI-Prototyping Tool namens *moqups*[1] zurückgegriffen, um die interaktiven Mockups zu erstellen. Obwohl die online gefundenen Mockup Tools wie *moqups* für die Erstellung von Webseiten ausgelegt sind, gab es bei der Umsetzung der beiden Ansätze keine Probleme. Jede Ansicht der beiden Ansätze entspricht dabei einer einzelnen Seite. Viele der benötigten Elemente wie beispielsweise geometrische Formen, Standard-Fenster und -Tabs und einige weitere waren dabei schon vorgegeben und konnten über Drag and Drop einfach auf die Seite gezogen werden. Die notwendigen Interaktionen innerhalb der Ansätze wurden über Verlinkungen zwischen den Seiten gelöst. Die folgenden Abbildungen 8.2 - 8.10 zeigen Ausschnitte der erstellten Mockups.

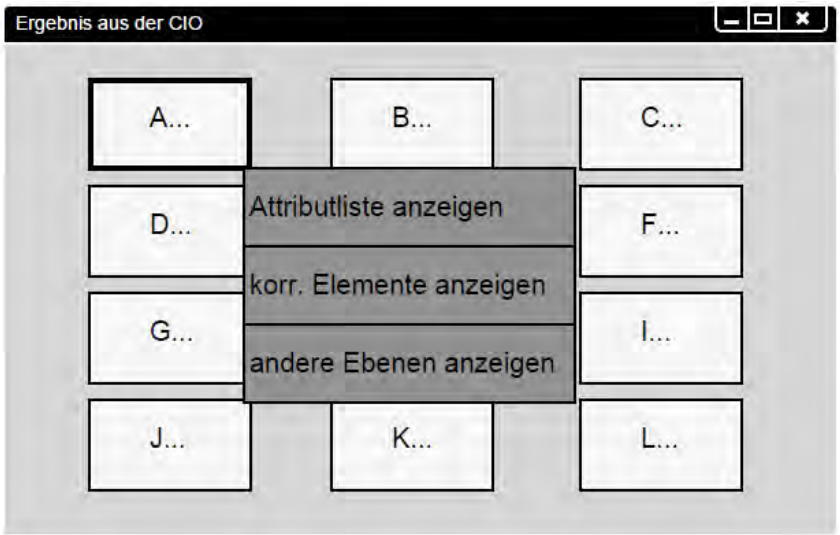


Abbildung 8.2: Ansatz 1: Nach Klick auf ein Element öffnet sich das Funktionsmenü.

[1] https://moqups.com

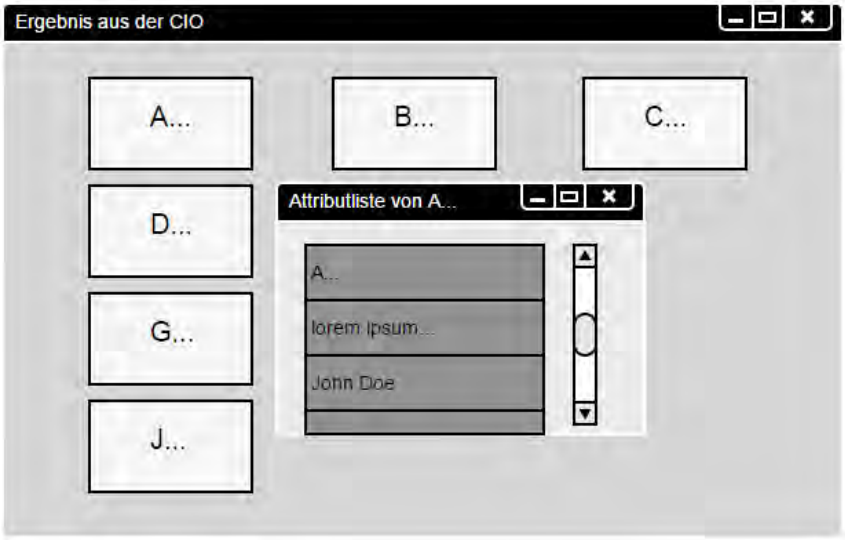


Abbildung 8.3: Ansatz 1: Die Attributliste eines Elements wird in einem eigenem Fenster angezeigt.

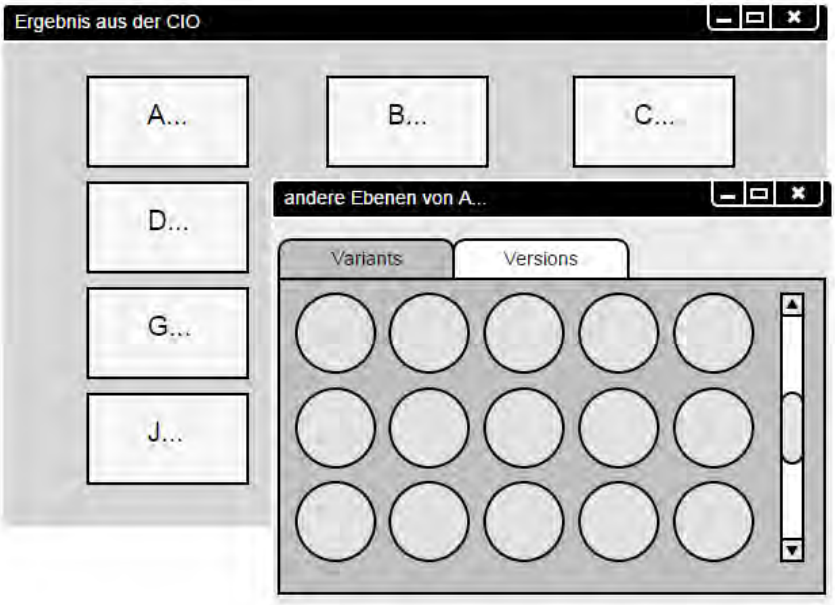


Abbildung 8.4: Ansatz 1: Auch die Varianten und Versionen lassen sich in einzelnen Tabs umsetzen.

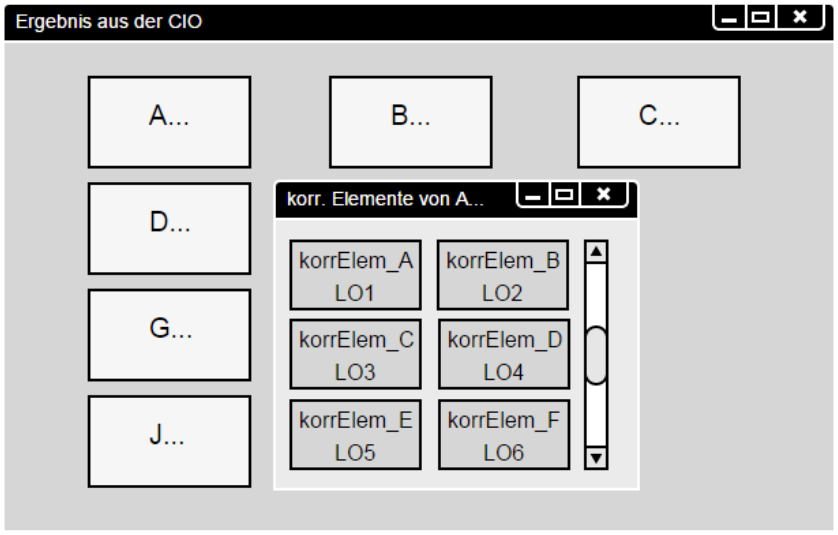


Abbildung 8.5: Ansatz 1: Die korrespondierenden Elemente finden ebenfalls in einem eigenen Fenster Platz.

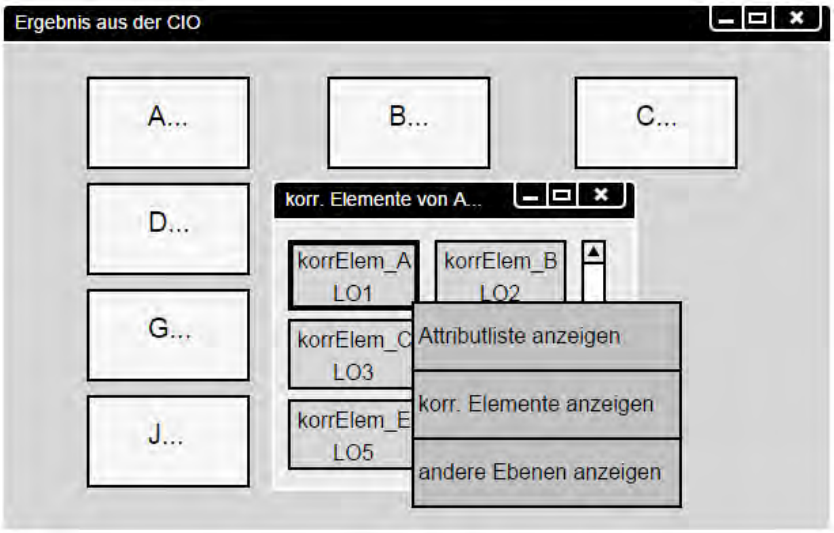


Abbildung 8.6: Ansatz 1: Nähere Informationen zu einem korrespondierendem Elemente erlangt man über das Funktionsmenü.

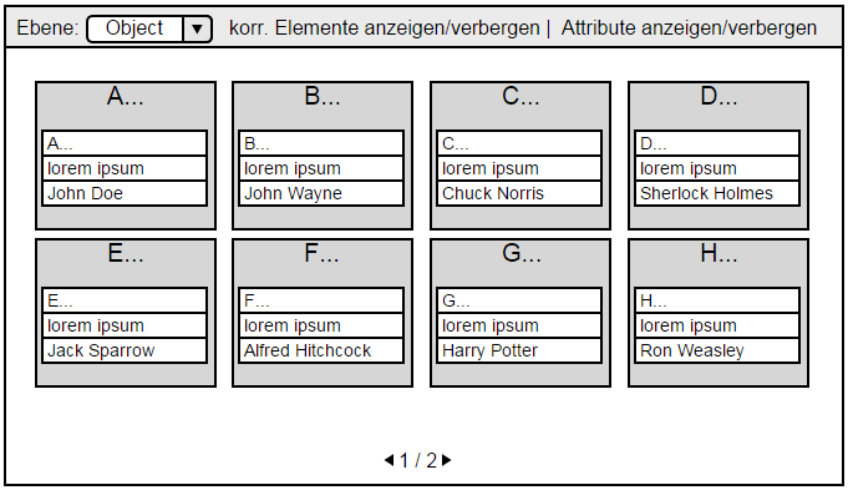


Abbildung 8.7: Ansatz 2: Die Startansicht der Ergebnismenge mit Attributliste, die sich aber auch verbergen lässt.

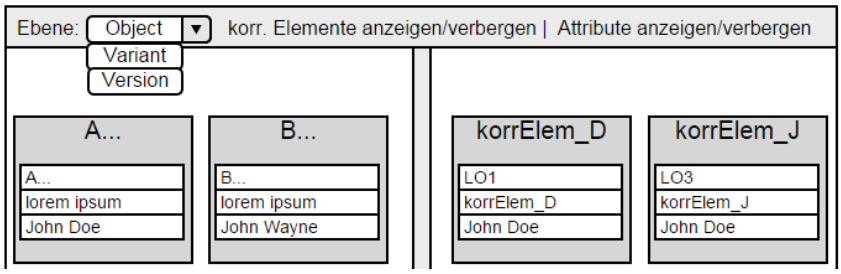


Abbildung 8.8: Ansatz 2: Ein Dropwdown-Menü bietet die Auswahl der verschiedenen Datenebenen.

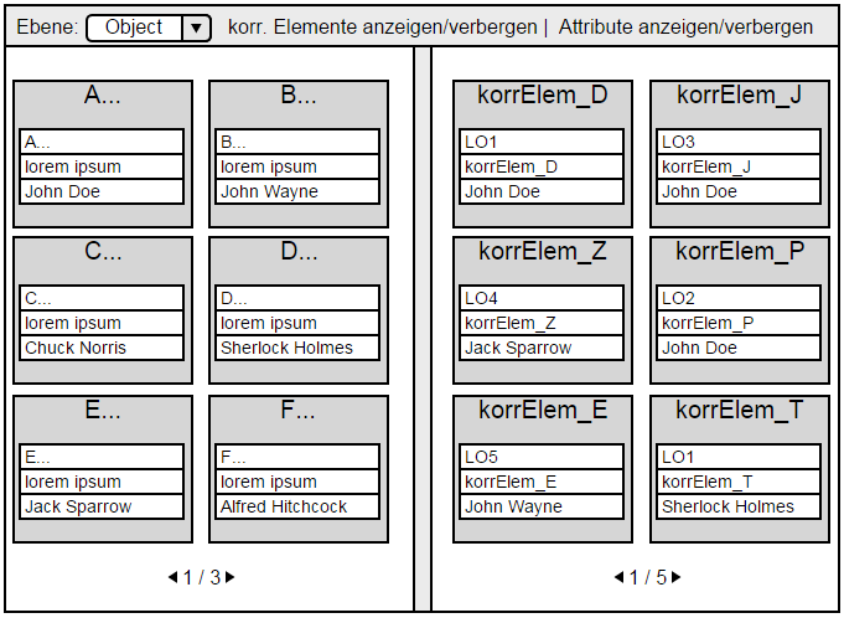


Abbildung 8.9: Ansatz 2: Ansicht der Ergebniselemente links und der korrespondierenden Elemente rechts.

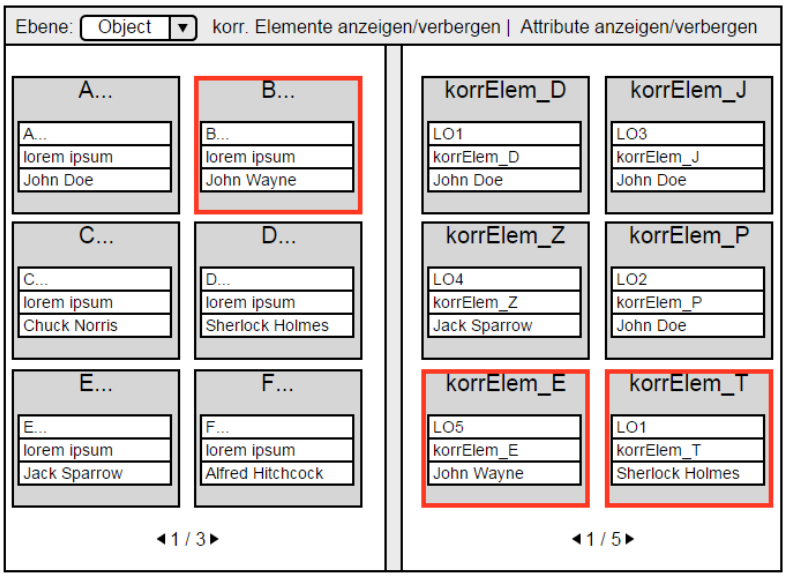


Abbildung 8.10: Ansatz 2: Farbliche Markierungen zeigen die Zusammengehörigkeit eines Ergebniselements und dessen korrespondierenden Elementen an.

## 8.2 Evaluationen und Tests

Durch die Erstellung der elektronischen, interaktiven Mockups ist gezeigt, dass sich die in dieser Arbeit entwickelten Ansätze auch technisch umsetzen lassen. Somit ist im Rahmen der Visualisierung und Navigation von Produktdaten nur noch ein Abgleich mit den Benutzern durchzuführen. Dieser ist zwar für ein ergonomisch hochwertiges Softwaresystem von essentieller Bedeutung, konnte jedoch auf Grund von Zeit und fehlenden Nutzern für diese Arbeit nicht durchgeführt werden. Nichtsdestotrotz unterlag die Entwicklung der Ansätze dem Referenzmodell für Usability Engineering, wodurch auf einige Kriterien des Usability Engineerings Rücksicht genommen wurde. Somit ist auch trotz fehlender Benutzereinbeziehung eine benutzerfreundliche beziehungsweise benutzbare Umgebung für die Visualisierung und Navigation von Produktdaten entstanden.

# Zusammenfassung

Technischer Fortschritt in der Automobilbranche bedeutet gleichzeitig eine steigende Anzahl von elektrischen und elektronischen Produktdaten. Da diese teils gleichen Daten von verschiedenen Nutzern auf unterschiedliche Weise in unterschiedlichen Applikationen bezeichnet und dokumentiert werden, sind die Datenmodelle heterogen. Dieses Problem soll mit Hilfe des PROCEED-Frameworks gelöst werden, indem die Produktdaten mit diesem Framework in die einheitliche Sicht, die Common Integration Ontology, integriert werden. Vorab jedoch müssen die Produktdaten erst noch in Local Ontologies transformiert werden, um alle Daten einheitlich auf die vier Produktdatenebenen Product Data Collection, Object, Variant und Version zu bringen. Auf Basis der CIO können dann Datenabfragen realisiert werden, deren Ergebnisse durch die Erkenntnisse und Entwicklungen dieser Arbeit visuell dargestellt werden können. Bevor jedoch auf die eigentliche Entwicklung eingegangen werden konnte, mussten sowohl grundlegende psychologische Theorien und die Bedeutungen und Funktionen von Bildern als auch das Konzept des Usability Engineerings nähergebracht werden (vgl. Kapitel 3). Die Usability ist ein nicht zu unterschätzender Aspekt in der heutigen Softwareentwicklung, bei der unentwegt die späteren Benutzer miteinbezogen werden. Außerdem definiert die Normreihe 9241 unter anderem einige Grundsätze bezüglich der Gestaltung von Dialogsystemen, an denen sich ebenfalls in dieser Arbeit orientiert wurde. Als roter Faden bei der Berücksichtigung der Usability der Visualisierung und Navigation von Produktdaten im PROCEED-Framework diente das Referenzmodell für Usability Engineering der Daimler AG. Von diesem wurden die für diese Arbeit relevanten Phasen und Prozessschritte genannt und jede einzelne davon analysiert. Innerhalb der Analysephase entstanden dabei entsprechende Anforderungen vom Framework (vgl. Kapitel 4), die in Kapitel 5 an die Gestaltung übertragen wurden. Dabei ist vor allem eine rechteckige Darstellungsform für die einzelnen Elemente und eine mehrspaltige Liste als Gesamtdarstellung als vorteilhaft herausgekommen. Als Darstellungsrahmen kamen von Beginn an Fenster oder Tabs in Frage, da sich diese beiden Ansätze in typischen Softwareanwendungen an Computern

durchgesetzt haben. Eine zentrale Unterscheidung, die oftmals in der Arbeit getätigt werden musste, war die Wahl zwischen einer elementweisen oder globalen Herangehensweise bei der Darstellung oder Navigation innerhalb der Ergebnisse. Beide Punkte bringen sowohl bestimmte Vor- und Nachteile mit sich, die im jeweiligen Fall abgewogen werden mussten. Diese Unterscheidung trat hauptsächlich bei der Berücksichtigung der Darstellung der korrespondierenden Elemente und beim Ebenenwechsel innerhalb der Ergebnisanzeige auf. Auf Basis all der gewonnenen Erkenntnisse in diesen Kapiteln wurde in den Kapiteln 6 und 7 ein kompletter elementweiser und ein kompletter globaler visueller Ansatz entwickelt. Dabei wurde bei der jeweiligen Entwicklung durchgehend Bezug auf die analysierten Anforderungen genommen. Um die technische Umsetzbarkeit der beiden Ansätze zu überprüfen, wurden mit Hilfe eines Online Mockup und Wireframe Tools elektronische UI-Prototypen der beiden Ansätze entwickelt. Dadurch lässt sich schlussfolgernd sagen, dass trotz fehlender Benutzerevaluationen zwei Ansätze entstanden sind, die aufgrund der Berücksichtigung psychologischer und benutzerfreundlicher Aspekte sowie einer ausführlichen Anforderungsanalyse durchaus für eine Realisierung innerhalb des PROCEED-Frameworks in Frage kommen.

# Abbildungsverzeichnis


2.1 Datenheterogenität . . . . . 10
2.2 Beispiel einer Local Ontology mit allen dazugehörigen Produktdatenebenen und beispielhaften Schema-Concepts und Individuals. . . . . 12
2.3 Beispiel von LO und CIO mit eingezeichneten Korrespondenzen . . . . . 13

3.1 Modell des integrierten Text- und Bildverstehens . . . . . 19
3.2 Arbeitsgedächtniskapazität . . . . . 21
3.3 Schalenmodell . . . . . 28
3.4 Typischer Systementwicklungsansatz . . . . . 30
3.5 Therapieorientierter Entwicklungsansatz . . . . . 31
3.6 Referenzmodell der Daimler AG . . . . . 32
3.7 Relevante Phasen und Prozessschritte dieser Arbeit . . . . . 34

4.1 Relevante Prozessschritte Analysephase . . . . . 38
4.2 CIO-Anzeige mit Beispielergebnis und Beispielelement . . . . . 40
4.3 Attributverlust bei Integration von LO nach CIO . . . . . 41

5.1 Relevante Prozessschritte des User-Interface-Entwurfs . . . . . 45
5.2 2D- bzw. 3D-Ansicht der Ergebniselemente . . . . . 46
5.3 Darstellungsmöglichkeiten der Artefaktattribute . . . . . 48
5.4 Flächeninhalts-Diagramm . . . . . 53
5.5 Ungeordnete Darstellung . . . . . 54
5.6 Seitenansicht . . . . . 57
5.7 Auslagerung der korr. Elemente in extra Fenster . . . . . 58
5.8 Auslagerung der korr. Elemente in neues Tab . . . . . 59
5.9 Verwendung von Tooltips . . . . . 60
5.10 Möglichkeiten der Darstellung der korr. Elemente mit oder ohne LO . . . . . 61
5.11 Eigenes Fenster für jede LO . . . . . 62

5.12 Ein Fenster mit jeweils einem Tab pro LO . . . . . 63
5.13 Ein Fenster mit einer Anzeige für korr. Elemente . . . . . 64
5.14 Auslagerung in neues Fenster nach Klick . . . . . 65
5.15 Auslagerung in neues Tab nach Klick . . . . . 65
5.16 Elementweises Fenster nach Klick im Funktionsmenü . . . . . 66
5.17 Elementweises Fenster nach Klick im Funktionsmenü . . . . . 67
5.18 Beispiele verschiedener Darstellungsarten von Zusammengehörigkeit . . . . 68
5.19 Farbliche Zusammengehörigkeit . . . . . 69
5.20 Verschiedene Formen für Objekte, Varianten und Versionen. . . . . 71
5.21 Eigenes Fenster für jede Datenebene . . . . . 73
5.22 Ein Fenster für beide Ebenen . . . . . 74
5.23 Verwendung von einzelnen Tabs pro Elementebene. . . . . 75
5.24 Seitenansicht der verschiedenen Elementebenen . . . . . 76
5.25 Darstellung einer globalen Schaltfläche . . . . . 77
5.26 Zuordnungsproblem beim globalen Ebenenwechsel . . . . . 78
5.27 Aktivierte globale Schaltfläche nach Klick . . . . . 79
5.28 Elementweiser Ebenenwechsel im Funktionsmenü . . . . . 80
5.29 Darstellung der Varianten in neuem Fenster . . . . . 80
5.30 Alternative Darstellung beider Ebenen . . . . . 81
5.31 Korr. Elemente in eigenem Fenster mit drei Tabs . . . . . 82
5.32 Verwendung einzelner Tabs für korrespondierende Ebenen . . . . . 83

6.1 Form der einzelnen Objekte . . . . . 85
6.2 Klick auf Objekt öffnet elementweises Funktionsmenü . . . . . 86
6.3 Funktion zur Anzeige der elementweisen Attributliste . . . . . 87
6.4 Elementweises Fenster mit Attributliste . . . . . 87
6.5 Elem. Ansatz: Rechteckige Gesamtdarstellung . . . . . 88
6.6 Elem. Ansatz: Fenster als Rahmen . . . . . 89
6.7 Elem. Ansatz: Fenster für korr. Elemente . . . . . 90
6.8 Elem. Ansatz: Infos der korr. Elemente . . . . . 91
6.9 Elem. Ansatz: erweitertes Funktionsmenü . . . . . 92
6.10 Elem. Ansatz: Klick auf Element öffnet Funktionsmenü . . . . . 93
6.11 Elem. Ansatz: Klick auf Schaltfläche für korr. Elemente . . . . . 93
6.12 Elem. Ansatz: geöffnetes Fenster für korr. Elemente . . . . . 94

6.13 Elem. Ansatz: 1:n Beziehung . . . . . 95
6.14 Elem. Ansatz: n:1 Beziehung . . . . . 95
6.15 Elem. Ansatz: verschwendeter Darstellungsplatz . . . . . 97
6.16 Elem. Ansatz: Fenster für andere Ebenen . . . . . 98
6.17 Elem. Ansatz: endgültiges Funktionsmenü . . . . . 98
6.18 Elem. Ansatz: Fenster plus Tabs für Ebenen . . . . . 99

7.1 Glob. Ansatz: Darstellung einzelner Elemente . . . . . 102
7.2 Glob. Ansatz: Rahmen der Elemente . . . . . 103
7.3 Glob. Ansatz: Darstellungsort korr. Elemente . . . . . 105
7.4 Glob. Ansatz: Darstellung korr. Elemente . . . . . 106
7.5 Glob. Ansatz: Gesamtdarstellung korr. Elemente . . . . . 107
7.6 Glob. Ansatz: Schaltfläche für korr. Elemente . . . . . 108
7.7 Glob. Ansatz: Anzeige korr. Elemente 1 . . . . . 109
7.8 Glob. Ansatz: Anzeige korr. Elemente 2 . . . . . 109
7.9 Glob. Ansatz: Anzeige korr. Elemente 3 . . . . . 110
7.10 Glob. Ansatz: Zusammengehörigkeit . . . . . 111
7.11 Glob. Ansatz: Ebenenwechsel Dropdown-Menü . . . . . 112
7.12 Glob. Ansatz: Beispiel Varianten-Auswahl . . . . . 113

8.1 Elektronische UI-Prototypen und Phase Evaluationen und Tests . . . . . 115
8.2 Ansatz 1: Funktionsmenü . . . . . 116
8.3 Ansatz 1: Attributliste . . . . . 117
8.4 Ansatz 1: Varianten und Versionen . . . . . 117
8.5 Ansatz 1: Korrespondierende Elemente . . . . . 118
8.6 Ansatz 1: Funktionsmenü bei korr. Elementen . . . . . 118
8.7 Ansatz 2: Ergebnismenge mit Attributen . . . . . 119
8.8 Ansatz 2: Dropwdown-Menü für Ebenenwechsel . . . . . 119
8.9 Ansatz 2: Ansicht der korr. Elemente . . . . . 120
8.10 Ansatz 2: Zusammengehörigkeit . . . . . 120

# Literaturverzeichnis


[CS91] CHANDLER, Paul ; SWELLER, John: Cognitive Load Theory and the Format of Instruction. In: *Cognition and Instruction* 8 (1991), S. 293–332

[Dzi83] DZIDA, Wolfgang: Das IFIP-Modell für Benutzerschnittstellen. In: *Office Management* 31, Sonderheft (1983), S. 6–8

[FGH+09] FISCHER, Richard ; GSCHEIDLE, Rolf ; HEIDER, Uwe ; HOHMANN, Berthold ; KEIL, Wolfgang ; MANN, Jochen ; SCHLÖGL, Bernd ; WIMMER, Alois ; WORMER, Günter: *Fachkunde Kraftfahrzeugtechnik*. Verlag Europa-Lehrmittel, 2009

[ITW] ITWISSEN, Online-Lexikon: *Software Ergonomie Usability Engineering*. `http://www.itwissen.info/definition/lexikon/Software-Ergonomie-usability-engineering.html`, Abruf: 21.01.2015

[Kar13] KARGL, Prof. Dr. Frank: *Skript: Fortgeschrittene Konzepte der Rechnernetze 03 - Automotive and Industrial Networks, Universität Ulm*. 2013

[May01] MAYER, Richard E.: *Multimedia Learning*. Cambridge University Press, 2001

[Off13a] OFFERGELD, Michael: *Skript: Usability Engineering 01 - Grundmodelle - Schalenmodell, Universität Ulm*. 2013

[Off13b] OFFERGELD, Michael: *Skript: Usability Engineering 01 - Grundmodelle 1, Universität Ulm*. 2013

[Off13c] OFFERGELD, Michael: *Skript: Usability Engineering 01 - Grundmodelle 2, Universität Ulm*. 2013

[Off13d] OFFERGELD, Michael: *Skript: Usability Engineering 02 - Projektvorbereitung, Universität Ulm*. 2013

[Off13e] OFFERGELD, Michael: *Skript: Usability Engineering 03 - Anforderungsanalyse, Universität Ulm*. 2013

[Off13f] OFFERGELD, Michael: *Skript: Usability Engineering 04 - UI Entwurf, Universität Ulm*. 2013

[Off13g] OFFERGELD, Michael: *Skript: Usability Engineering 05 - Restphasen, Universität Ulm*. 2013

[Pai86] PAIVIO, Allan: *Mental representations: a dual coding approach*. Oxford University Press, 1986

[RF13] RICHTER, Michael ; FLÜCKIGER, Markus D.: *Usability Engineering kompakt*. Springer Vieweg, 2013

[SB99] SCHNOTZ, Wolfgang ; BANNERT, Maria: Einflüsse der Visualisierungsform auf die Konstruktion mentaler Modelle beim Text- und Bildverstehen. In: *Experimental Psychology* 46 (1999), S. 217–236

[Seu13a] SEUFERT, Prof. Dr. Tina: *Skript: Medienpsychologie und -pädagogik 03_Medienspezifische Lerntheorien, Universität Ulm*. 2013

[Seu13b] SEUFERT, Prof. Dr. Tina: *Skript: Medienpsychologie und -pädagogik 06_Bildgestaltung, Universität Ulm*. 2013

[Seu13c] SEUFERT, Prof. Dr. Tina: *Skript: Medienpsychologie und -pädagogik 07_Multimediagestaltung, Universität Ulm*. 2013

[The94] THE STANDISH GROUP: The CHAOS Report (1994) / The Standish Group International Inc. Version: 1994. `http://www.standishgroup.com/sample_research_files/chaos_report_1994.pdf`. 1994. – Forschungsbericht

[THR13] TIEDEKEN, Julian ; HERBST, Joachim ; REICHERT, Manfred: Managing Complex Data for Electrical/Electronic Components: Challenges and Requirements. In:

*Proc. BTW'13 Workshops, 15th Conf on Database Systems, Technology, and Web (BTW'13)*. Magdeburg, March 2013

[TRH13] TIEDEKEN, Julian ; REICHERT, Manfred ; HERBST, Joachim: On the Integration of Electrical/Electronic Product Data in the Automotive Domain. In: *Datenbank Spektrum* 13 (2013), S. 189–199

[Wika] WIKIPEDIA: *Digital Mock-Up*. http://de.wikipedia.org/w/index.php?title=Digital_Mock-Up&oldid=135756626, Abruf: 30.01.2015

[Wikb] WIKIPEDIA: *Physical Mock-Up*. http://de.wikipedia.org/w/index.php?title=Physical_Mock-Up&oldid=83387044, Abruf: 30.01.2015

[Wikc] WIKIPEDIA: *Reihenfolge*. http://de.wikipedia.org/w/index.php?title=Reihenfolge&oldid=111603011, Abruf: 28.12.2014

[Win14] WINFUTURE: *Windows Store hat nun 150.000 Apps im Angebot*. http://winfuture.de/news,80884.html. Version: März 2014, Abruf: 28.12.2014

Name: Stefan Hausladen Matrikelnummer: 750801

**Erklärung**

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Stefan Hausladen